

# Proiect – Baze de Date

## Gestiunea unui lanț de biblioteci

Ganea Alexandru Gabriel

Grupa 211

# Cuprins

1. Descrierea modelului real
2. Prezentarea constrângerilor
3. Descrierea entităților
4. Descrierea relațiilor
5. Descrierea atributelor
6. Realizarea diagramei entitate-relație
7. Realizarea diagramei conceptuale
8. Enumerarea schemelor relaționale
9. Realizarea normalizării până la forma normală 3
10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele
11. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea
12. Cereri SQL complexe
13. Operații de actualizare și de suprimare a datelor utilizând subcereri
14. Crearea unei vizualizări complexe
15. Cereri SQL, outer-join, division și top-n
16. Comparția a două instrucțiuni echivalente semantic
17. a. Realizarea normalizării BCNF, FN4, FN5.  
    b. Relații/Join-uri din model reprezentate într-o bază de date nosql
18. Tranzacții: ilustrarea consistency levels
19. Optimizarea a două cereri utilizând indexare

# 1. Descrierea modelului real

Baza de date este utilizată pentru gestionarea unui lanț de biblioteci. Bibliotecile se află în diferite locații și se cunosc diferite date despre acestea. Pe lângă locația și informațiile despre biblioteci, baza de date va conține datele despre fiecare carte disponibilă cât și despre editura acestora, autorii și domeniul din care fac parte. Fiecare cititor poate avea un abonament sau poate face împrumuturi. De asemenea, pe lângă aceste informații, baza de date conține și informații despre fiecare angajat al unei biblioteci.

## Utilitatea modelului

Pentru un lanț mare de biblioteci este necesară o bază de date pentru a gestiona mai ușor și mai eficient toate informațiile despre bibliotecă precum toate cărțile din toate bibliotecile, toate împrumuturile făcute de diferiți cititori precum și date despre fiecare bibliotecă în parte și angajații acestora.

## Reguli de funcționare

Modelul proiectat respectă anumite reguli de funcționare. Fiecare bibliotecă se află într-o singură locație și are cel puțin un angajat. Fiecare cititor poate face câte împrumuturi dorește atâta timp cât are un abonament valid, se va restricționa adăugarea unui împrumut dacă această condiție nu se respectă. Cu toate că un cititor poate face câte împrumuturi dorește, fiecare carte împrumutată se va trece ca un împrumut nou. Fiecare carte trebuie să aibă un autor și un domeniu din care face parte, dacă numărul de exemplare pentru o carte dintr-o anumită bibliotecă este 0, nu se va putea face împrumutul din acea bibliotecă.

## 2. Prezentarea constrângerilor

Pentru modelul proiectat există următoarele constrângeri:

- ◆ O bibliotecă se află într-o singură locație
- ◆ Într-o locație exista doar o bibliotecă
- ◆ O bibliotecă poate avea mai mulți angajați dar cel puțin 1
- ◆ Un angajat lucrează într-o singură bibliotecă
- ◆ O carte poate avea mai mulți autori
- ◆ Un autor poate scrie mai multe cărți
- ◆ O carte are o singură editură
- ◆ O editură poate publica mai multe cărți
- ◆ O carte aparține unui singur domeniu
- ◆ Un domeniu conține mai multe cărți
- ◆ O bibliotecă are mai multe cărți
- ◆ Un împrumut poate împrumuta mai multe cărți de la biblioteci diferite
- ◆ Un împrumut se poate realiza doar dacă cititorul are abonamentul activ
- ◆ Un cititor poate avea doar un abonament
- ◆ Un abonament aparține unui cititor

### 3. Descrierea entităților

Modelul proiectat cuprinde următoarele entități:

- **CARTE**, care conține informații despre cărțile din diferite biblioteci. Cheia primară a acestei entități este *#id\_carte*.
- **EDITURA**, care conține informații despre diferite edituri. Cheia primară a acestei entități este *#id\_editura*.
- **DOMENIU**, care conține denumirea unui domeniu din care face parte o carte. Cheia primară a acestei entități este *#id\_domeniu*.
- **AUTOR**, care conține detalii despre diferiți autori. Cheia primară a acestei entități este *#id\_autor*.
- **CITITOR**, care conține informații despre fiecare cititor. Cheia primară a acestei entități este *#id\_cititor*.
- **ABONAMENT**, care conține detalii despre un abonament al unui cititor. Cheia primară a acestei entități este *#id\_abonament*.
- **BIBLIOTECA**, care conține informații despre o bibliotecă. Cheia primară a acestei entități este *#id\_biblioteca*.
- **LOCATIE**, care conține detalii despre fiecare locație a unei biblioteci. Cheia primară a acestei entități este *#id\_locatie*.
- **ANGAJAT**, care conține informații despre fiecare angajat al unei biblioteci. Cheia primară a acestei entități este *#id\_angajat*.
- **IMPRUMUT**, care conține fiecare informație necesară unui împrumut de carte al unui cititor dintr-o bibliotecă. Cheia primară a acestei entități este *#id\_imprumut*.

## 4. Descrierea relațiilor

În continuare voi prezenta toate relațiile între entități, o mică descriere a lor cât și cardinalitatea acestora.

**BIBLIOTECA-se\_afla-LOCATIE** = Relația leagă entitățile **BIBLIOTECA** și **LOCATIE**, relația semnifică faptul că o bibliotecă se află într-o anumită locație. Cardinalitatea este 1:1.

**ANGAJAT-lucreeza-BIBLIOTECA** = Relația leagă entitățile **ANGAJAT** și **BIBLIOTECA** și semnifică faptul că un angajat lucrează într-o bibliotecă. Cardinalitatea este M:1.

**BIBLIOTECA-contine-CARTE** = Relația leagă entitățile **BIBLIOTECA** și **CARTE** și semnifică faptul că o bibliotecă conține mai multe cărți. Cardinalitatea este M:N.

**AUTOR-scrie-CARTE** = Relația leagă entitățile **AUTOR** și **CARTE** și semnifică faptul că un autor scrie o carte. Cardinalitatea este M:N.

**EDITURA-publica-CARTE** = Relația leagă entitățile **EDITURA** și **CARTE** și semnifică faptul că o editură publică o carte. Cardinalitatea este 1:M.

**DOMENIU-contine-CARTE** = Relația leagă entitățile **DOMENIU** și **CARTE** și semnifică faptul că un domeniu conține o carte. Cardinalitatea este 1:M.

**CITITOR-publica-ABONAMENT** = Relația leagă entitățile **CITITOR** și **ABONAMENT** și semnifică faptul că un cititor poate avea un abonament. Cardinalitatea este 1:1.

**BIBLIOTECA-realizeaza-IMPRUMUT** = Relația leagă entitățile **BIBLIOTECA** și **IMPRUMUT** și semnifică faptul că o bibliotecă realizează un împrumut. Cardinalitatea este 1:M.

**CITITOR-realizeaza-IMPRUMUT** = Relația leagă entitățile **CITITOR** și **IMPRUMUT** și semnifică faptul că un cititor realizează un împrumut. Cardinalitatea este 1:M.

**IMPRUMUT-contine-CARTE** = Relația leagă entitățile **IMPRUMUT** și **CARTE** și semnifică faptul că un împrumut conține o carte. Cardinalitatea este M:1.

## 5. Descrierea atributelor

Pentru fiecare din entități voi prezenta atributele și eventuale constrângeri. Descrierea propriu-zisă a fiecărui atribut nu este necesară deoarece fiecare dintre ele are un nume sugestiv.

Entitatea **LOCATIE** conține:

*#id\_locatie*= cheia primară, reprezentând id-ul locației, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*oraș*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*strada*= variabilă de tip șir de caractere de lungime maximă 30.

Entitatea **BIBLIOTECA** conține:

*#id\_biblioteca*= cheia primară, reprezentând id-ul bibliotecii, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*denumire*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*id\_locatie*= variabilă de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **LOCATIE** și **BIBLIOTECA**.

Entitatea **ANGAJAT** conține:

*#id\_angajat*= cheia primară, reprezentând id-ul unui angajat, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*nume*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*prenume*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*nr\_tel*= variabilă de tip șir de caractere de lungime maximă 10.

*functie*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*salariu*= variabilă de tip întreg de maxim 5 cifre.

*data\_angajarii*= variabilă de tip dată, nu poate fi nulă.

*id\_biblioteca*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **ANGAJAT** și **BIBLIOTECA**.

Entitatea **CITITOR** conține:

*#id\_cititor*= cheia primară, reprezentând id-ul unui cititor, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*nume*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*prenume*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*nr\_tel*= variabilă de tip șir de caractere de lungime maximă 10.

*data\_nasterii*= variabilă de tip dată.,

Entitatea **ABONAMENT** conține:

*#id\_abonament*= cheia primară, reprezentând id-ul unui abonament, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*pret*= variabilă de tip întreg de maxim 5 cifre, nu poate fi nulă.

*data\_obtinerii*= variabilă de tip dată, nu poate fi nulă.

*data\_expirare*= variabilă de tip dată, nu poate fi nulă.

*id\_cititor*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **ABONAMENT** și **CITITOR**.

Entitatea **DOMENIU** conține:

*#id\_domeniu*= cheia primară, reprezentând id-ul unui domeniu, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*denumire*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

Entitatea **EDITURA** conține:

*#id\_editura*= cheia primară, reprezentând id-ul unei edituri, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*denumire*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*sediu*= variabila de tip șir de caractere de lungime maximă 20.

*an\_aparitiei*= variabilă de tip întreg de maxim 4 cifre.

Entitatea **CARTE** conține:

*#id\_carte*= cheia primară, reprezentând id-ul unei cărți, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*denumire*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*nr\_pagini*= variabilă de tip întreg de maxim 5 cifre.



*an\_aparitie*= variabilă de tip întreg de maxim 4 cifre.

*pret*= variabilă de tip întreg de maxim 5 cifre, nu poate fi nulă.

*id\_editura*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **CARTE** și **EDITURA**.

*id\_domeniu*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **CARTE** și **DOMENIU**.

Entitatea **AUTOR** conține:

*#id\_autor*= cheia primară, reprezentând id-ul unui autor, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*nume*= variabilă de tip șir de caractere de lungime maximă 20, nu poate fi nulă.

*prenume*= variabilă de tip șir de caractere de lungime maximă 20.

*data\_nasterii*= variabilă de tip dată.

Tabelul asociativ **CARTE\_AUTOR** conține:

*#id\_carte*= face parte din cheia primară împreună cu *#id\_autor* reprezintă id-ul unei cărți, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă. Reprezintă și legătura dintre entitățile **CARTE\_AUTOR** și **CARTE**.

*#id\_autor*= face parte din cheia primară împreună cu *#id\_carte*, reprezintă id-ul unei cărți, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă. Reprezintă și legătura dintre entitățile **CARTE\_AUTOR** și **CARTE**.

Tabelul asociativ **CARTE\_BIBLIOTECA** conține:

*#id\_carte*= face parte din cheia primară împreună cu *#id\_biblioteca* reprezintă id-ul unei cărți, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă. Reprezintă și legătura dintre entitățile **CARTE\_BIBLIOTECA** și **CARTE**.

*#id\_biblioteca*= face parte din cheia primară împreună cu *#id\_carte*, reprezintă id-ul unei cărți, variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă. Reprezintă și legătura dintre entitățile **CARTE\_BIBLIOTECA** și **BIBLIOTECA**.

*nr\_exemplare*= variabilă de tip întreg de maxim 4 cifre, nu poate fi nulă.

Entitatea **IMPRUMUT** conține:

*#id\_imprumut*= cheia primară, reprezentând id-ul unui împrumut variabilă de tip întreg de maxim 6 cifre, nu poate fi nulă.

*data\_imprumut*= variabilă de tip dată, nu poate fi nulă.

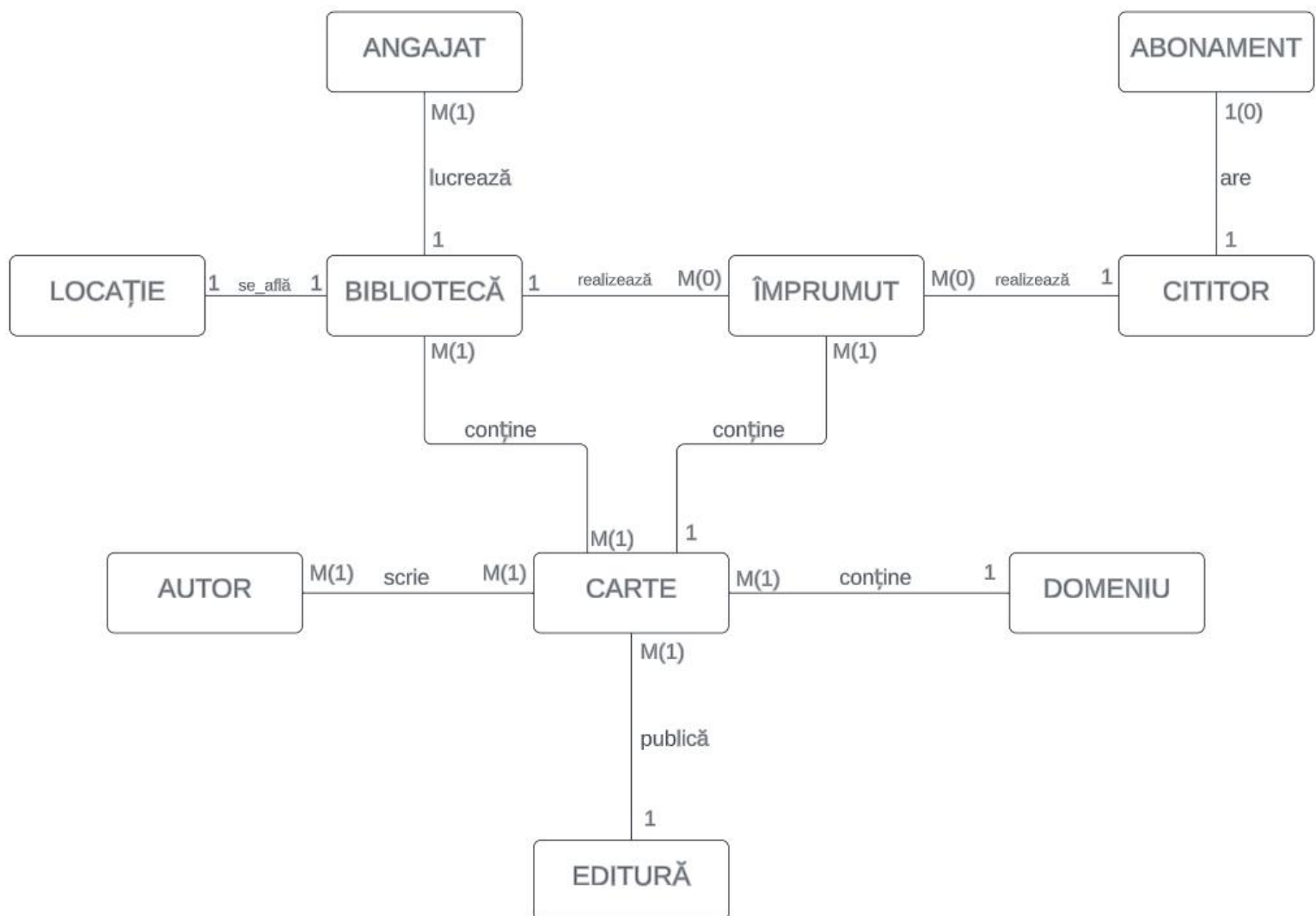
*data\_restituire*= variabilă de tip dată, nu poate fi nulă.

*id\_biblioteca*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **IMPRUMUT** și **BIBLIOTECA**.

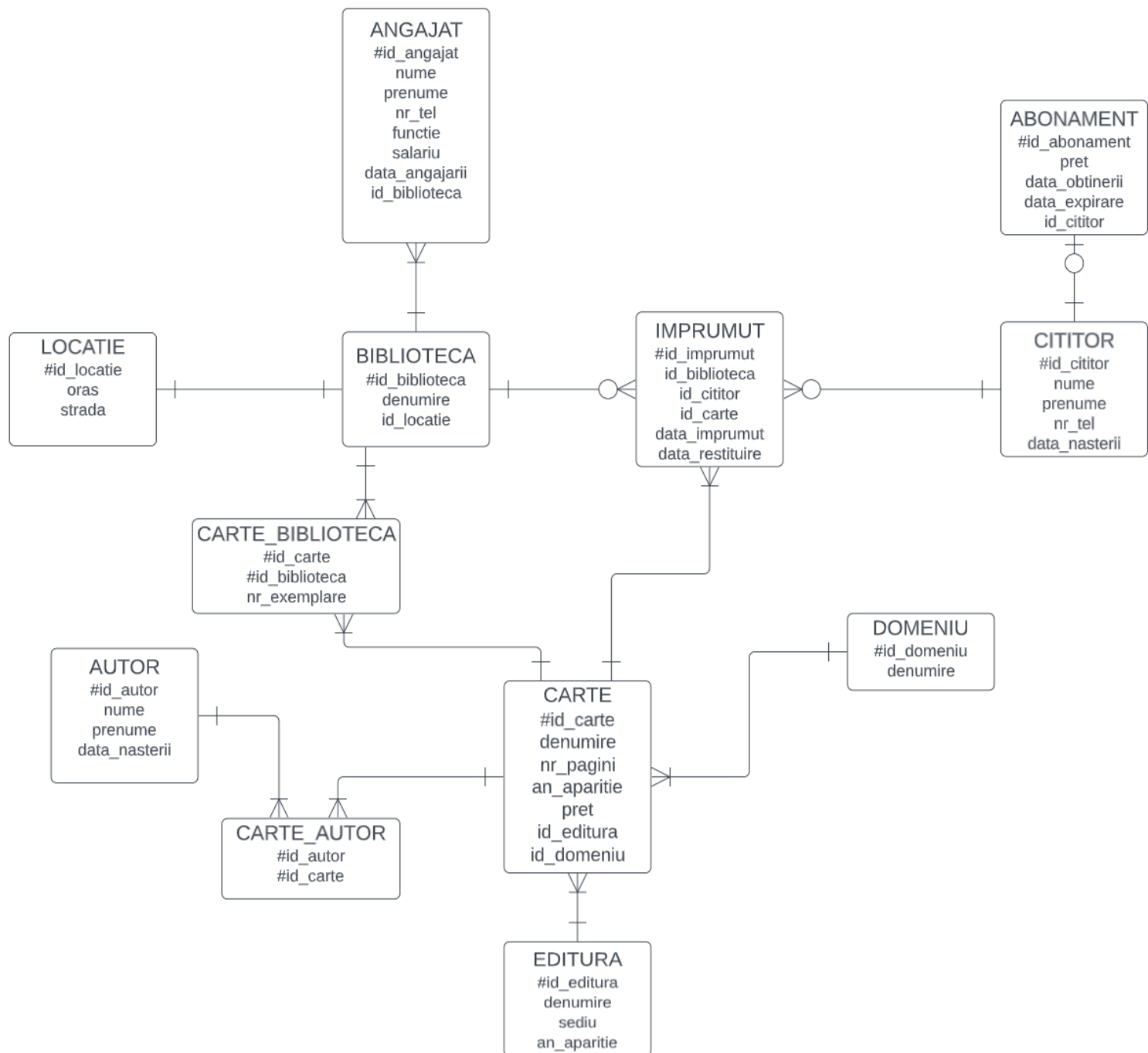
*id\_cititor*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **IMPRUMUT** și **CITITOR**.

*id\_carte*= variabila de tip întreg de maxim 6 cifre. Reprezintă legătura dintre entitățile **IMPRUMUT** și **CARTE**.

## 6. Diagrama entitate-relație



## 7. Diagrama conceptuală



## 8. Enumerarea schemelor relaționale

LOCATIE (#id\_locatie, oras, strada)

BIBLIOTECA (#id\_biblioteca, denumire, id\_locatie)

ANGAJAT (#id\_angajat, nume, prenume, nr\_tel, functie, salariu, data\_angajarii, id\_biblioteca)

CITITOR (#id\_cititor, nume, prenume, nr\_tel, data\_nasterii)

ABONAMENT (#id\_abonament, pret, data\_obtinerii, data\_expirare, id\_cititor)

DOMENIU (#id\_domeniu, denumire)

EDITURA (#id\_editura, denumire, sediu, an\_aparitie)

CARTE (#id\_carte, denumire, nr\_pagini, an\_aparitie, pret, id\_editura, id\_domeniu)

AUTOR (#id\_autor, nume, prenume, data\_nasterii)

CARTE\_AUTOR (#id\_autor, #id\_carte)

CARTE\_BIBLIOTECA (#id\_carte, #id\_biblioteca, nr\_exemplare)

IMPRUMUT (#id\_imprumut, id\_biblioteca, id\_cititor, id\_carte, data\_imprumut, data\_restituire)

## 9. Realizarea normalizărilor până la forma normală 3

### • Forma FN1

O relație se află în forma normală 1 (FN1) dacă și numai dacă, toate attributele sale iau numai valori atomice. O bază de date relaționară trebuie să fie cel puțin în forma normală 1. Putem observa că modelul proiectat se află în FN1 deoarece fiecărui atribut îi corespunde o valoare atomică.

Un exemplu care n-ar fi în FN1 ar putea apărea, de exemplu, la unul din tabelele asociative precum **CARTE\_BIBLIOTECA**.

ID_BIBLIOTECA	ID_CARTE
B1	C1,C2
B2	C3,C4,C5
B3	C2

-exemplu non FN1-

Putem observa că tabelul de mai sus îl putem aduce în forma normală 1 dacă împărțim coloana CARTE pe linii diferite pentru fiecare valoare.

ID_BIBLIOTECA	ID_CARTE
B1	C1
B1	C2
B2	C3
B2	C4
B2	C5
B3	C2

-exemplu FN1-

## • Forma FN2

Modelul de date proiectat se află în forma normală 2 deoarece se află în forma normală 1 și fiecare atribut care nu este cheie, depinde de întreaga cheie primară.

Pentru a exemplifica, vom lua același tabel ca mai sus, adăugând attributele *PRET* și *NR\_EXEMPLARE*.

ID_BIBLIOTECA	ID_CARTE	PRET	NR_EXEMPLARE
B1	C1	20	10
B1	C2	30	15
B2	C2	30	13
B3	C3	15	20

-exemplu non FN2-

Având cheia formată din biblioteca *id\_biblioteca* și *id\_carte*, putem observa că atributul non-cheie *PRET* depinde doar de *id\_carte* ceea ce face ca tabelul să nu fie în FN2. Pentru a aduce modelul în forma normală 2, este necesar să împărțim tabelul în două tabele.

ID_BIBLIOTECA	ID_CARTE	NR_EXEMPLARE
B1	C1	10
B1	C2	15
B2	C2	13
B3	C3	20

ID_CARTE	PRET
C1	20
C2	30
C3	15

-exemplu FN2-

## • Forma FN3

Modelul de date proiectat se află în forma normală 3 deoarece se află în forma normală 2 și fiecare atribut care nu este cheie, depinde de cheie și numai de cheie.

Pentru a exemplifica, vom lua tabelul **CARTE** ca exemplu care ar putea să nu fie în FN3 și îl vom normaliza. Pentru acest exemplu, vom folosi *id\_carte* ca cheie primară, iar ca non-chei avem attributele *pret*, *id\_editura* și *denumire\_editura*.

ID_CARTE	PRET	ID_EDITURA	DENUMIRE_EDITURA
C1	20	E1	DE1
C2	30	E2	DE2
C3	15	E3	DE3

-exemplu non FN3-

Observăm că atributul *denumire\_editura* este independent de *id\_carte* care reprezintă cheia primară în tabelul nostru ceea ce îl face să nu fie în forma normală 3. Deoarece *denumire\_editura* depinde de *id\_editura* pentru a trece tabelul în FN3 este necesar să despărțim tabelul în două tabele.

ID_CARTE	PRET	ID_EDITURA
C1	20	E1
C2	30	E2
C3	15	E3

ID_EDITURA	DENUMIRE_EDITURA
E1	DE1
E2	DE2
E3	DE3

-exemplu FN3-

## 10. Crearea unor secvențe ce vor fi folosite la inserare

Pentru o inserare mai eficientă a datelor, pentru tabelele cu multe înregistrări voi folosi niște secvențe care vor fi semnifica id-urile acestor tabele.

```
CREATE SEQUENCE SEQ_CARTE  
MAXVALUE 9999  
START WITH 1  
INCREMENT BY 1;
```

```
CREATE SEQUENCE SEQ_CITITOR  
MAXVALUE 9999  
START WITH 1  
INCREMENT BY 1;
```

```
CREATE SEQUENCE SEQ_ANGAJAT  
MAXVALUE 9999  
START WITH 1  
INCREMENT BY 1;
```

```
CREATE SEQUENCE SEQ_AUTOR  
MAXVALUE 9999  
START WITH 1  
INCREMENT BY 1;
```



## 11. Crearea tabelelor în SQL și inserarea de date

- Crearea și inserarea în tabelul **LOCATIE**

```
create table locatie(  
    id_locatie number(6),  
    oras varchar2(20) not null,  
    strada varchar2(30),  
    primary key (id_locatie)  
);
```

```
insert into locatie values(1, 'Bucuresti', 'Bulevardul unirii 22');  
insert into locatie values(2, 'Bucuresti', 'Strada veteranilor 7-9');  
insert into locatie values(3, 'Bucuresti', 'Str. Boteanu nr. 1');  
insert into locatie values(4, 'Cluj', 'Str. Clinicilor 2');  
insert into locatie values(5, 'Cluj', 'Mihail Kogălniceanu 12-14');  
insert into locatie values(6, 'Ploiesti', 'Erou Călin Cătălin 1');  
insert into locatie values(7, 'Pitesti', 'Victoriei 18');
```

	ID_LOCATIE	ORAS	STRADA
1	1	Bucuresti	Bulevardul unirii 22
2	2	Bucuresti	Strada veteranilor 7-9
3	3	Bucuresti	Str. Boteanu nr. 1
4	4	Cluj	Str. Clinicilor 2
5	5	Cluj	Mihail Kogălniceanu 12...
6	6	Ploiesti	Erou Călin Cătălin 1
7	7	Pitesti	Victoriei 18

-printscreen sql-developer-

- Crearea si inserarea în tabelul **BIBLIOTECA**

```
create table biblioteca(
    id_biblioteca number(6),
    denumire varchar2(40) not null,
    id_locatie number(6),
    primary key (id_biblioteca),
    constraint fk_biblioteca foreign key (id_locatie) references
locatie(id_locatie)
);
```

```
insert into biblioteca values(1, 'Biblioteca Națională a României', 1);
insert into biblioteca values(2, 'Biblioteca Metropolitană București', 2);
insert into biblioteca values(3, 'Biblioteca Centrală Universitară', 3);
insert into biblioteca values(4, 'Biblioteca Lucian Blaga', 4);
insert into biblioteca values(5, 'Biblioteca Județeană Octavian Goga', 5);
insert into biblioteca values(6, 'Biblioteca Județeană Nicolae Iorga', 6);
insert into biblioteca values(7, 'Biblioteca Județeană Dinicu Golescu', 7);
```

ID_BIBLIOTECA	DENUMIRE	ID_LOCATIE
1	1Biblioteca Națională a României	1
2	2Biblioteca Metropolitană București	2
3	3Biblioteca Centrală Universitară	3
4	4Biblioteca Lucian Blaga	4
5	5Biblioteca Județeană Octavian Goga	5
6	6Biblioteca Județeană Nicolae Iorga	6
7	7Biblioteca Județeană Dinicu Gole...	7

-screenshot sql-developer-

- Crearea si inserarea în tabelul **AUTOR**

```
create table autor(  
    id_autor number(6),  
    nume varchar2(20) not null,  
    prenume varchar2(20),  
    data_nasterii date,  
    primary key (id_autor)  
);
```

```
CREATE SEQUENCE SEQ_AUTOR  
MAXVALUE 9999  
START WITH 1  
INCREMENT BY 1;
```

```
insert into autor values (seq_autor.nextval, 'Mihai' , 'Eminescu',  
to_date('15/01/1850','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'Stephen' , 'King',  
to_date('21/09/1947','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'William' , 'Shakespeare',  
to_date('23/04/1564','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'George' , 'Orwell',  
to_date('25/06/1903','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'Ioan' , 'Slavici',  
to_date('18/01/1848','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'Ion' , 'Creangă',  
to_date('01/03/1837','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'Sarah' , 'Mass',  
to_date('05/03/1986','dd/mm/yyyy'));  
insert into autor values (seq_autor.nextval, 'Clare' , 'Cassandra',  
to_date('27/07/1973','dd/mm/yyyy'));
```

insert into autor values (seq\_autor.nextval, 'Dostoyevsky' , 'Fyodor',  
to\_date('11/11/1821','dd/mm/yyyy'));

insert into autor values (seq\_autor.nextval, 'Camil' , 'Petrescu',  
to\_date('22/04/1894','dd/mm/yyyy'));

	ID_AUTOR	NUME	PRENUME	DATA_NASTERII
1	1	Mihai	Eminescu	15-JAN-50
2	2	Stephen	King	21-SEP-47
3	3	William	Shakespeare	23-APR-64
4	4	George	Orwell	25-JUN-03
5	5	Ioan	Slavici	18-JAN-48
6	6	Ion	Creangă	01-MAR-37
7	7	Sarah	Mass	05-MAR-86
8	8	Clare	Cassandra	27-JUL-73
9	9	Dostoyevsky	Fyodor	11-NOV-21
10	10	Camil	Petrescu	22-APR-94

-screenshot sql-developer-

- Crearea si inserarea în tabelul **DOMENIU**

create table domeniu(

id\_domeniu number(6),

denumire varchar2(20) not null,

primary key (id\_domeniu)

);

insert into domeniu values (1,'Literatura romana');

insert into domeniu values (2,'Literatura straina');

insert into domeniu values (3,'Fictiune');

insert into domeniu values (4,'Filosofie');

insert into domeniu values (5,'Diverse');

	ID_DOMENIU	DENUMIRE
1	1	Literatura romana
2	2	Literatura straina
3	3	Fictiune
4	4	Filosofie
5	5	Diverse

-printscreen sql-developer-

- Crearea si inserarea în tabelul **EDITURA**

create table editura(

id\_editura number(6),  
denumire varchar2(20) not null,  
sediu varchar2(20),  
an\_aparitie number(4),  
primary key (id\_editura)

);

insert into editura values (1, 'Litera', 'Bucuresti', 1989);

insert into editura values (2, 'Humanitas', 'Bucuresti', 1990);

insert into editura values (3, 'Trei', 'Bucuresti', 1995);

insert into editura values (4, 'Corint', 'Bucuresti', 1994);

insert into editura values (5, 'Paralela 45', 'Bucuresti', 1994);

	ID_EDITURA	DENUMIRE	SEDIU	AN_APARITIE
1	1	Litera	Bucuresti	1989
2	2	Humanitas	Bucuresti	1990
3	3	Trei	Bucuresti	1995
4	4	Corint	Bucuresti	1994
5	5	Paralela 45	Bucuresti	1994

-printscreen sql-developer-

- Crearea si inserarea în tabelul **CARTE**

```
create table carte(  
    id_carte number(6),  
    denumire varchar2(40) not null,  
    nr_pagini number(5),  
    an_aparitie number(4),  
    pret number(5) not null,  
    id_editura number(6),  
    id_domeniu number(6),  
    primary key (id_carte),  
    constraint fk_carte_editura foreign key (id_editura) references  
editura(id_editura),  
    constraint fk_carte_domeniu foreign key (id_domeniu) references  
domeniu(id_domeniu)  
);
```

```
CREATE SEQUENCE SEQ_CARTE  
MAXVALUE 9999  
START WITH 1  
INCREMENT BY 1;
```

```
insert into carte values(seq_carte.nextval, 'Furtuna',32,1611,20,1,2);  
insert into carte values(seq_carte.nextval, 'Hamlet',104,1603,35,1,2);  
insert into carte values(seq_carte.nextval, 'Macbeth',122,1606,20,2,2);  
insert into carte values(seq_carte.nextval, 'Romeo si  
Julieta',136,1595,25,3,2);  
insert into carte values(seq_carte.nextval, '11/22/63',849,2011,50,4,2);  
insert into carte values(seq_carte.nextval, 'Cimitirul  
animalelor',496,1983,35,4,2);  
insert into carte values(seq_carte.nextval, 'Luceafarul',60,1883,10,3,1);  
insert into carte values(seq_carte.nextval, 'Poesii',307,1883,35,3,1);
```

```

insert into carte values(seq_carte.nextval, '1984',320,1949,35,5,2);
insert into carte values(seq_carte.nextval, 'Ferma
animalelor',86,1945,25,5,2);
insert into carte values(seq_carte.nextval, 'Amintiri din
copilarie',332,1892,15,2,1);
insert into carte values(seq_carte.nextval, 'Povestea lui Harap-
Alb',168,1877,10,2,1);
insert into carte values(seq_carte.nextval, 'Moara cu
noroc',160,1881,12,1,1);
insert into carte values(seq_carte.nextval, 'Tronul de
clestar',512,2012,30,4,3);
insert into carte values(seq_carte.nextval, 'Taramul de
cenusa',992,2018,45,4,3);
insert into carte values(seq_carte.nextval, 'Mostenitoarea
Focului',656,2014,30,4,3);
insert into carte values(seq_carte.nextval, 'Stapanul
umbrelor',656,2017,35,2,3);
insert into carte values(seq_carte.nextval, 'Instrumente
mortale',432,2011,45,2,3);
insert into carte values(seq_carte.nextval, 'Crima si
pedeapsa',720,1866,50,1,4);
insert into carte values(seq_carte.nextval, 'Fratii
Karamazov',840,1880,50,1,4);
insert into carte values(seq_carte.nextval, 'Jocul
ielelor',169,1916,38,5,1);
insert into carte values(seq_carte.nextval, 'Patul lui
Procust',336,1933,45,5,1);

```

ID_CARTE	DENUMIRE	NR_PAGINI	AN_APARITIE	PRET	ID_EDITURA	ID_DOMENIU
1	1 Furtuna	32	1611	20	1	2
2	2 Hamlet	104	1603	35	1	2
3	3 Macbeth	122	1606	20	2	2
4	4 Romeo si Julieta	136	1595	25	3	2
5	5 11/22/63	849	2011	50	4	2
6	6 Cimitirul animalelor	496	1983	35	4	2
7	7 Luceafarul	60	1883	10	3	1
8	8 Poesii	307	1883	35	3	1
9	9 1984	320	1949	35	5	2
10	10 Ferma animalelor	86	1945	25	5	2
11	11 Amintiri din copila...	332	1892	15	2	1
12	12 Povestea lui Harap-...	168	1877	10	2	1

-printscreen sql-developer-

- Crearea si inserarea în tabelul **CARTE\_AUTOR**

```
create table carte_autor(  
    id_autor number(6),  
    id_carte number(6),  
    primary key (id_autor, id_carte),  
    constraint fk_carte_autor_id_autor foreign key (id_autor) references  
    autor(id_autor),  
    constraint fk_carte_autor_id_carte foreign key (id_carte) references  
    carte(id_carte)  
);
```

```
insert into carte_autor values(3,1);  
insert into carte_autor values(3,2);  
insert into carte_autor values(3,3);  
insert into carte_autor values(3,4);  
insert into carte_autor values(2,5);  
insert into carte_autor values(2,6);  
insert into carte_autor values(1,7);  
insert into carte_autor values(1,8);  
insert into carte_autor values(4,9);  
insert into carte_autor values(4,10);  
insert into carte_autor values(6,11);  
insert into carte_autor values(6,12);  
insert into carte_autor values(5,13);  
insert into carte_autor values(7,14);  
insert into carte_autor values(7,15);  
insert into carte_autor values(7,16);  
insert into carte_autor values(8,17);  
insert into carte_autor values(8,18);
```



```

insert into carte_autor values(9,19);
insert into carte_autor values(9,20);
insert into carte_autor values(10,21);
insert into carte_autor values(10,22);

```

	ID_AUTOR	ID_CARTE
1	1	7
2	1	8
3	2	5
4	2	6
5	3	1
6	3	2

-printscreen sql-developer-

- Crearea si inserarea în tabelul **CARTE\_BIBLIOTECA**

```

create table carte_biblioteca(
    id_carte number(6),
    id_biblioteca number(6),
    nr_exemplare number(5) not null,
    primary key (id_carte, id_biblioteca),
    constraint fk_carte_biblioteca_id_carte foreign key (id_carte)
references carte(id_carte),
    constraint fk_carte_biblioteca_id_biblioteca foreign key
(id_biblioteca) references biblioteca(id_biblioteca)
);

```

```

insert into carte_biblioteca values(1,1,6);
insert into carte_biblioteca values(2,1,3);
insert into carte_biblioteca values(3,1,7);
insert into carte_biblioteca values(4,1,5);

```

```
insert into carte_biblioteca values(5,1,4);
insert into carte_biblioteca values(6,1,7);
insert into carte_biblioteca values(7,1,4);
insert into carte_biblioteca values(8,1,5);
insert into carte_biblioteca values(9,1,3);
insert into carte_biblioteca values(10,1,7);
insert into carte_biblioteca values(11,1,3);
insert into carte_biblioteca values(12,1,2);
insert into carte_biblioteca values(13,1,5);
insert into carte_biblioteca values(14,1,1);
insert into carte_biblioteca values(15,1,3);
insert into carte_biblioteca values(16,1,4);
insert into carte_biblioteca values(17,1,2);
insert into carte_biblioteca values(18,1,6);
insert into carte_biblioteca values(19,1,5);
insert into carte_biblioteca values(20,1,3);
insert into carte_biblioteca values(21,1,1);
insert into carte_biblioteca values(22,1,4);
insert into carte_biblioteca values(1,2,12);
insert into carte_biblioteca values(3,2,2);
insert into carte_biblioteca values(5,2,1);
insert into carte_biblioteca values(7,2,4);
insert into carte_biblioteca values(9,2,7);
insert into carte_biblioteca values(11,2,9);
insert into carte_biblioteca values(13,2,10);
insert into carte_biblioteca values(2,3,21);
insert into carte_biblioteca values(4,3,2);
insert into carte_biblioteca values(6,3,2);
insert into carte_biblioteca values(8,3,3);
insert into carte_biblioteca values(10,3,5);
insert into carte_biblioteca values(12,3,2);
```

```
insert into carte_biblioteca values(14,3,4);
insert into carte_biblioteca values(20,4,1);
insert into carte_biblioteca values(13,4,5);
insert into carte_biblioteca values(14,4,6);
insert into carte_biblioteca values(2,4,8);
insert into carte_biblioteca values(4,4,9);
insert into carte_biblioteca values(7,4,2);
insert into carte_biblioteca values(19,4,11);
insert into carte_biblioteca values(22,5,3);
insert into carte_biblioteca values(21,5,2);
insert into carte_biblioteca values(15,5,4);
insert into carte_biblioteca values(17,5,2);
insert into carte_biblioteca values(19,5,5);
insert into carte_biblioteca values(3,5,9);
insert into carte_biblioteca values(2,5,12);
insert into carte_biblioteca values(14,6,2);
insert into carte_biblioteca values(5,6,2);
insert into carte_biblioteca values(13,6,3);
insert into carte_biblioteca values(12,6,6);
insert into carte_biblioteca values(10,6,4);
insert into carte_biblioteca values(21,6,6);
insert into carte_biblioteca values(1,6,12);
insert into carte_biblioteca values(1,7,10);
insert into carte_biblioteca values(2,7,5);
insert into carte_biblioteca values(22,7,3);
insert into carte_biblioteca values(21,7,4);
insert into carte_biblioteca values(13,7,2);
insert into carte_biblioteca values(15,7,1);
insert into carte_biblioteca values(9,7,5);
```

	ID_CARTE	ID_BIBLIOTECA	NR_EXEMPLARE
1	1	1	6
2	2	1	3
3	3	1	7
4	4	1	5
5	5	1	4
6	6	1	7
7	7	1	4

-printscreen sql-developer-

- Crearea si inserarea în tabelul **ANGAJAT**

create table angajat(

id\_angajat number(6),

nume varchar2(20) not null,

prenume varchar2(20) not null,

nr\_tel varchar2(10),

functie varchar2(20) not null,

salariu number(5),

data\_angajarii date not null,

id\_biblioteca number(6),

primary key (id\_angajat),

constraint fk\_angajat foreign key (id\_biblioteca) references  
biblioteca(id\_biblioteca)

);

insert into angajat values(seq\_angajat.nextval, 'Tomescu', 'Pavel',  
'0724322453', 'Bibliotecar Sef', 4500  
,to\_date('30/01/2022','dd/mm/yyyy'), 1);

insert into angajat values(seq\_angajat.nextval, 'Popescu', 'Maria',  
'0755123456', 'Asistent Bibliotecar', 3000, to\_date('10/03/2018',  
'dd/mm/yyyy'), 1);

insert into angajat values(seq\_angajat.nextval, 'Ionescu', 'Andrei',  
'0766789123', 'Receptie', 3200, to\_date('05/07/2017', 'dd/mm/yyyy'),  
1);

```

insert into angajat values(seq_angajat.nextval, 'Georgescu', 'Ana',
'0727567890', 'Asistent Bibliotecar', 2800, to_date('18/09/2015',
'dd/mm/yyyy'), 1);

insert into angajat values(seq_angajat.nextval, 'Constantinescu',
'Alexandru', '0746234501', 'Arhivar', 3100, to_date('22/11/2020',
'dd/mm/yyyy'), 1);

insert into angajat values(seq_angajat.nextval, 'Mihai', 'Cristina',
'0737987654', 'Asistent Bibliotecar', 2900, to_date('14/02/2023',
'dd/mm/yyyy'), 1);

insert into angajat values(seq_angajat.nextval, 'Chirita', 'Madalina',
'0773243192', 'Bibliotecar Sef', 4500
,to_date('15/04/2021','dd/mm/yyyy'), 2);

insert into angajat values(seq_angajat.nextval, 'Dumitrescu', 'Ion',
'0745123401', 'Asistent Bibliotecar', 3000, to_date('10/03/2018',
'dd/mm/yyyy'), 2);

insert into angajat values(seq_angajat.nextval, 'Stanescu', 'Mirela',
'0736789567', 'Receptie', 3200, to_date('05/07/2017', 'dd/mm/yyyy'),
2);

insert into angajat values(seq_angajat.nextval, 'Munteanu', 'Lucian',
'0727567123', 'Asistent Bibliotecar', 2800, to_date('18/09/2015',
'dd/mm/yyyy'), 2);

insert into angajat values(seq_angajat.nextval, 'Gheorghe', 'Andreea',
'0756234123', 'Arhivar', 3100, to_date('22/11/2020', 'dd/mm/yyyy'), 2);

insert into angajat values(seq_angajat.nextval, 'Popa', 'Mariana',
'0737987465', 'Asistent Bibliotecar', 2900, to_date('14/02/2023',
'dd/mm/yyyy'), 2);

insert into angajat values(seq_angajat.nextval, 'Popa', 'Emilian',
'0743228551', 'Bibliotecar Sef', 4500
,to_date('12/05/2023','dd/mm/yyyy'), 3);

insert into angajat values(seq_angajat.nextval, 'Popescu', 'Cristian',
'0745123433', 'Asistent Bibliotecar', 3000, to_date('10/03/2018',
'dd/mm/yyyy'), 3);

insert into angajat values(seq_angajat.nextval, 'Mihnea', 'Cristina',
'0736789563', 'Arhivar', 3200, to_date('05/07/2017', 'dd/mm/yyyy'), 3);

insert into angajat values(seq_angajat.nextval, 'Georgescu', 'Ana',
'0747567123', 'Asistent Bibliotecar', 2800, to_date('18/09/2015',
'dd/mm/yyyy'), 3);

```

```

insert into angajat values(seq_angajat.nextval, 'Constantin', 'Alexandru',
'0756234563', 'Arhivar', 3100, to_date('22/11/2020', 'dd/mm/yyyy'), 3);

insert into angajat values(seq_angajat.nextval, 'Mihai', 'Ionel',
'0727987653', 'Asistent Bibliotecar', 2900, to_date('14/02/2023',
'dd/mm/yyyy'), 3);

insert into angajat values(seq_angajat.nextval, 'Mocanu', 'Victor',
'0733123643', 'Bibliotecar Sef', 4500
,to_date('14/12/2022','dd/mm/yyyy'), 4);

insert into angajat values(seq_angajat.nextval, 'Dumitrescu', 'Ana',
'0735123444', 'Asistent Bibliotecar', 3000, to_date('10/03/2018',
'dd/mm/yyyy'), 4);

insert into angajat values(seq_angajat.nextval, 'Stanescu', 'Andreea',
'0756789124', 'Arhivar', 3200, to_date('05/07/2017', 'dd/mm/yyyy'), 4);

insert into angajat values(seq_angajat.nextval, 'Munteanu', 'Radu',
'0747567344', 'Asistent Bibliotecar', 2800, to_date('18/09/2015',
'dd/mm/yyyy'), 4);

insert into angajat values(seq_angajat.nextval, 'Gheorghe', 'Elena',
'0726234544', 'Receptie', 3100, to_date('22/11/2020', 'dd/mm/yyyy'),
4);

insert into angajat values(seq_angajat.nextval, 'Popa', 'Catalin',
'0737987644', 'Asistent Bibliotecar', 2900, to_date('14/02/2023',
'dd/mm/yyyy'), 4);

insert into angajat values(seq_angajat.nextval, 'Vasilica', 'Denisa',
'0741278831', 'Bibliotecar Sef', 4500
,to_date('20/10/2021','dd/mm/yyyy'), 5);

insert into angajat values(seq_angajat.nextval, 'Vasilescu', 'Mihai',
'0725123455', 'Asistent Bibliotecar', 3000, to_date('02/06/2019',
'dd/mm/yyyy'), 5);

insert into angajat values(seq_angajat.nextval, 'Popovici', 'Andreea',
'0736789565', 'Receptie', 3200, to_date('07/09/2020', 'dd/mm/yyyy'),
5);

insert into angajat values(seq_angajat.nextval, 'Radulescu', 'Cristina',
'0747567155', 'Asistent Bibliotecar', 2800, to_date('14/03/2017',
'dd/mm/yyyy'), 5);

insert into angajat values(seq_angajat.nextval, 'Diaconu', 'Alexandru',
'0756234575', 'Arhivar', 3100, to_date('28/05/2018', 'dd/mm/yyyy'), 5);

```

```

insert into angajat values(seq_angajat.nextval, 'Gheorghiu', 'Elena',
'0737987655', 'Asistent Bibliotecar', 2900, to_date('19/11/2022',
'dd/mm/yyyy'), 5);

insert into angajat values(seq_angajat.nextval, 'Calinescu', 'Valentin',
'0724172642', 'Bibliotecar Sef', 4500
,to_date('13/05/2020','dd/mm/yyyy'), 6);

insert into angajat values(seq_angajat.nextval, 'Popescu', 'Adrian',
'0725123466', 'Asistent Bibliotecar', 3000, to_date('12/04/2022',
'dd/mm/yyyy'), 6);

insert into angajat values(seq_angajat.nextval, 'Ivanov', 'Maria',
'0736789566', 'Receptie', 3200, to_date('25/08/2019', 'dd/mm/yyyy'),
6);

insert into angajat values(seq_angajat.nextval, 'Dragoi', 'Ion',
'0747567166', 'Asistent Bibliotecar', 2800, to_date('07/06/2016',
'dd/mm/yyyy'), 6);

insert into angajat values(seq_angajat.nextval, 'Moraru', 'Gabriela',
'0756234576', 'Arhivar', 3100, to_date('19/09/2021', 'dd/mm/yyyy'), 6);

insert into angajat values(seq_angajat.nextval, 'Popoviciu', 'Alexandra',
'0737987656', 'Asistent Bibliotecar', 2900, to_date('02/02/2017',
'dd/mm/yyyy'), 6);

insert into angajat values(seq_angajat.nextval, 'Muresan', 'Sebastian',
'0720658521', 'Bibliotecar Sef', 4500
,to_date('05/03/2020','dd/mm/yyyy'), 7);

insert into angajat values(seq_angajat.nextval, 'Mihai', 'Roxana',
'0725123477', 'Asistent Bibliotecar', 3000, to_date('14/09/2022',
'dd/mm/yyyy'), 7);

insert into angajat values(seq_angajat.nextval, 'Stanescu', 'Cristian',
'0736789577', 'Receptie', 3200, to_date('27/12/2020', 'dd/mm/yyyy'),
7);

insert into angajat values(seq_angajat.nextval, 'Alexandrescu', 'Mihaela',
'0747567177', 'Asistent Bibliotecar', 2800, to_date('09/02/2019',
'dd/mm/yyyy'), 7);

insert into angajat values(seq_angajat.nextval, 'Popov', 'Adina',
'0756234577', 'Arhivar', 3100, to_date('21/05/2017', 'dd/mm/yyyy'), 7);

insert into angajat values(seq_angajat.nextval, 'Radulescu', 'Marius',
'0737987657', 'Asistent Bibliotecar', 2900, to_date('05/11/2023',
'dd/mm/yyyy'), 7);

```

ID_ANGAJAT	NUME	PRENUME	NR_TEL	FUNCTIE	SALARIU	DATA_ANGAJARII	ID_BIBLIOTECA
1	1 Tomescu	Pavel	0724322453	Bibliotecar Sef	4500	30-JAN-22	1
2	2 Popescu	Maria	0755123456	Asistent Biblio...	3000	10-MAR-18	1
3	3 Ionescu	Andrei	0766789123	Receptie	3200	05-JUL-17	1
4	4 Georgescu	Ana	0727567890	Asistent Biblio...	2800	18-SEP-15	1
5	5 Constant...	Alexandru	0746234501	Arhivar	3100	22-NOV-20	1
6	6 Mihai	Cristina	0737987654	Asistent Biblio...	2900	14-FEB-23	1
7	7 Chirita	Madalina	0773243192	Bibliotecar Sef	4500	15-APR-21	2
8	8 Dumitrescu	Ion	0745123401	Asistent Biblio...	3000	10-MAR-18	2
9	9 Stanescu	Mirela	0736789567	Receptie	3200	05-JUL-17	2
10	10 Munteanu	Lucian	0727567123	Asistent Biblio...	2800	18-SEP-15	2
11	11 Gheorghe	Andreea	0756234123	Arhivar	3100	22-NOV-20	2
12	12 Popa	Mariana	0737987465	Asistent Biblio...	2900	14-FEB-23	2
13	13 Popa	Emilian	0743228551	Bibliotecar Sef	4500	12-MAY-23	3
14	14 Popescu	Cristian	0745123433	Asistent Biblio...	3000	10-MAR-18	3
15	15 Mihnea	Cristina	0736789563	Arhivar	3200	05-JUL-17	3
16	16 Georgescu	Ana	0747567123	Asistent Biblio...	2800	18-SEP-15	3
17	17 Constantin	Alexandru	0756234563	Arhivar	3100	22-NOV-20	3
18	18 Mihai	Ionel	0727987653	Asistent Biblio...	2900	14-FEB-23	3
19	19 Mocanu	Victor	0733123643	Bibliotecar Sef	4500	14-DEC-22	4
20	20 Dumitrescu	Ana	0735123444	Asistent Biblio...	3000	10-MAR-18	4

-printscreensql-developer-

## • Crearea si inserarea în tabelul CITITOR

create table cititor(

id\_cititor number(6),  
 nume varchar2(20) not null,  
 prenume varchar2(20) not null,  
 nr\_tel varchar2(10),  
 data\_nasterii date,  
 primary key (id\_cititor)

);

insert into cititor values (seq\_cititor.nextval, 'Nae', 'Valentin',  
 '0720930442', to\_date('20/05/2002', 'dd/mm/yyyy'));

insert into cititor values(seq\_cititor.nextval, 'Popescu', 'Ana',  
 '0738123456', to\_date('17/01/2003', 'dd/mm/yyyy'));

insert into cititor values(seq\_cititor.nextval, 'Ionescu', 'Andrei',  
 '0756789123', to\_date('13/11/2004', 'dd/mm/yyyy'));

insert into cititor values(seq\_cititor.nextval, 'Georgescu', 'Maria',  
 '0729876543', to\_date('07/12/2005', 'dd/mm/yyyy'));

insert into cititor values(seq\_cititor.nextval, 'Constantinescu', 'Cristina',  
 '0732567891', to\_date('06/02/1987', 'dd/mm/yyyy'));



```

insert into cititor values(seq_cititor.nextval, 'Mihai', 'Alexandru',
'0745123789', to_date('12/03/1995', 'dd/mm/yyyy'));

insert into cititor values(seq_cititor.nextval, 'Popa', 'Ioana',
'0723489567', to_date('28/06/1993', 'dd/mm/yyyy'));

insert into cititor values(seq_cititor.nextval, 'Radu', 'Mihnea',
'0736745123', to_date('11/05/2002', 'dd/mm/yyyy'));

insert into cititor values(seq_cititor.nextval, 'Gheorghe', 'Elena',
'0742789561', to_date('15/12/1988', 'dd/mm/yyyy'));

insert into cititor values(seq_cititor.nextval, 'Stancu', 'Dragos',
'0758912345', to_date('20/10/2007', 'dd/mm/yyyy'));

```

	ID_CITITOR	NUME	PRENUME	NR_TEL	DATA_NASTERII
1	1	Nae	Valentin	0720930442	20-MAY-02
2	2	Popescu	Ana	0738123456	17-JAN-03
3	3	Ionescu	Andrei	0756789123	13-NOV-04
4	4	Georgescu	Maria	0729876543	07-DEC-05
5	5	Constantine...	Cristina	0732567891	06-FEB-87
6	6	Mihai	Alexandru	0745123789	12-MAR-95
7	7	Popa	Ioana	0723489567	28-JUN-93
8	8	Radu	Mihnea	0736745123	11-MAY-02
9	9	Gheorghe	Elena	0742789561	15-DEC-88
10	10	Stancu	Dragos	0758912345	20-OCT-07

-printscreen sql-developer-

- Crearea si inserarea în tabelul **ABONAMENT**

```

create table abonament(
    id_abonament number(6),
    pret number(5) not null,
    data_obtinerii date not null,
    data_expirare date not null,
    id_cititor number(6),
    primary key (id_abonament),

```

```

constraint fk_abonament foreign key (id_cititor) references
cititor(id_cititor)
);

```

```

insert into abonament values(1, 20, to_date('26/05/2023',
'dd/mm/yyyy'), to_date('10/12/2023', 'dd/mm/yyyy')+60, 1);
insert into abonament values(2, 100, to_date('27/05/2022',
'dd/mm/yyyy'), to_date('27/05/2022', 'dd/mm/yyyy')+365, 2);
insert into abonament values(7, 40, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+60, 3);
insert into abonament values(3, 100, to_date('15/03/2022',
'dd/mm/yyyy'), to_date('15/03/2022', 'dd/mm/yyyy')+365, 4);
insert into abonament values(4, 60, to_date('07/05/2023',
'dd/mm/yyyy'), to_date('07/07/2023', 'dd/mm/yyyy')+180, 5);
insert into abonament values(8, 60, to_date('07/05/2023',
'dd/mm/yyyy'), to_date('07/07/2023', 'dd/mm/yyyy')+180, 6);
insert into abonament values(5, 100, to_date('24/11/2022',
'dd/mm/yyyy'), to_date('24/11/2022', 'dd/mm/yyyy')+365, 9);
insert into abonament values(6, 60, to_date('06/10/2022',
'dd/mm/yyyy'), to_date('06/10/2022', 'dd/mm/yyyy')+180, 7);

```

	ID_ABONAMENT	PRET	DATA_OBTINERII	DATA_EXPIRARE	ID_CITITOR
1	1	20	26-MAY-23	08-FEB-24	1
2	2	100	27-MAY-22	27-MAY-23	2
3	7	40	20-APR-23	19-JUN-23	3
4	3	100	15-MAR-22	15-MAR-23	4
5	4	60	07-MAY-23	03-JAN-24	5
6	8	60	07-MAY-23	03-JAN-24	6
7	5	100	24-NOV-22	24-NOV-23	9
8	6	60	06-OCT-22	04-APR-23	7

-printscreen sql-developer-

- Crearea si inserarea în tabelul **IMPRUMUT**

```

create table imprumut(
    id_imprumut number(6),
    id_biblioteca number(6),
    id_cititor number(6),
    id_carte number(6),
    data_imprumut date not null,
    data_restituire date not null,
    primary key (id_imprumut),
    constraint fk_imprumut_id_biblioteca foreign key(id_biblioteca)
references biblioteca(id_biblioteca),
    constraint fk_imprumut_id_cititor foreign key(id_cititor) references
cititor(id_cititor),
    constraint fk_imprumut_id_carte foreign key(id_carte) references
carte(id_carte)
);

```

```

CREATE OR REPLACE TRIGGER trig_verifica_abonament
BEFORE INSERT OR UPDATE OF id_cititor ON imprumut
FOR EACH ROW
DECLARE
    x NUMBER;
    expirare_abonament DATE;
BEGIN
    SELECT COUNT(*) INTO x
    FROM abonament
    WHERE id_cititor = :new.id_cititor;

    IF x = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'CITITORUL NU ARE UN
ABONAMENT ACTIV PENTRU A FACE IMPRUMUTUL');

```

```

ELSE
    SELECT data_expirare INTO expirare_abonament
    FROM abonament
    WHERE id_cititor = :new.id_cititor;

    IF expirare_abonament < SYSDATE THEN
        RAISE_APPLICATION_ERROR(-20001, 'ABONAMENTUL CITITORULUI A
        EXPIRAT');
    END IF;
END IF;
END;

```

```

insert into imprumut values(1, 1, 1, 1, to_date('26/05/2023',
'dd/mm/yyyy'), to_date('26/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(2, 1, 1, 20, to_date('26/05/2023',
'dd/mm/yyyy'), to_date('26/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(3, 1, 1, 14, to_date('26/05/2023',
'dd/mm/yyyy'), to_date('26/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(4, 1, 1, 17, to_date('26/05/2023',
'dd/mm/yyyy'), to_date('26/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(5, 1, 1, 15, to_date('26/05/2023',
'dd/mm/yyyy'), to_date('26/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(6, 2, 3, 2, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(7, 2, 3, 3, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(8, 2, 3, 4, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(9, 2, 3, 4, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(10, 2, 3, 21, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+14);

```

```

insert into imprumut values(11, 2, 3, 22, to_date('20/04/2023',
'dd/mm/yyyy'), to_date('20/04/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(12, 3, 5, 12, to_date('07/05/2023',
'dd/mm/yyyy'), to_date('07/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(13, 3, 5, 16, to_date('10/05/2023',
'dd/mm/yyyy'), to_date('10/04/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(14, 3, 5, 5, to_date('12/05/2023',
'dd/mm/yyyy'), to_date('12/04/2023', 'dd/mm/yyyy')+10);
insert into imprumut values(15, 3, 5, 9, to_date('13/05/2023',
'dd/mm/yyyy'), to_date('13/04/2023', 'dd/mm/yyyy')+10);
insert into imprumut values(16, 1, 9, 5, to_date('24/11/2022',
'dd/mm/yyyy'), to_date('24/11/2022', 'dd/mm/yyyy')+14);
insert into imprumut values(17, 1, 9, 6, to_date('24/11/2022',
'dd/mm/yyyy'), to_date('24/11/2022', 'dd/mm/yyyy')+14);
insert into imprumut values(18, 5, 9, 8, to_date('04/03/2023',
'dd/mm/yyyy'), to_date('04/03/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(19, 5, 9, 9, to_date('04/03/2023',
'dd/mm/yyyy'), to_date('04/03/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(20, 5, 9, 13, to_date('04/03/2023',
'dd/mm/yyyy'), to_date('04/03/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(21, 7, 9, 14, to_date('24/05/2023',
'dd/mm/yyyy'), to_date('24/05/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(22, 4, 6, 10, to_date('07/05/2023',
'dd/mm/yyyy'),to_date('07/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(23, 4, 6, 11, to_date('07/05/2023',
'dd/mm/yyyy'),to_date('07/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(24, 4, 6, 12, to_date('07/05/2023',
'dd/mm/yyyy'),to_date('07/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(25, 4, 6, 15, to_date('07/05/2023',
'dd/mm/yyyy'),to_date('07/05/2023', 'dd/mm/yyyy')+7);
insert into imprumut values(26, 6, 6, 17, to_date('15/05/2023',
'dd/mm/yyyy'),to_date('15/05/2023', 'dd/mm/yyyy')+14);
insert into imprumut values(27, 6, 6, 7, to_date('15/05/2023',
'dd/mm/yyyy'),to_date('15/05/2023', 'dd/mm/yyyy')+14);

```

insert into imprumut values(28, 6, 6, 5, to\_date('15/05/2023',  
'dd/mm/yyyy'),to\_date('15/05/2023', 'dd/mm/yyyy')+14);

insert into imprumut values(29, 6, 6, 13, to\_date('15/05/2023',  
'dd/mm/yyyy'),to\_date('15/05/2023', 'dd/mm/yyyy')+14);

	ID_IMPRUMUT	ID_BIBLIOTECA	ID_CITITOR	ID_CARTE	DATA_IMPRUMUT	DATA_RESTITUIRE
1	1	1	1	1	26-MAY-23	02-JUN-23
2	2	1	1	20	26-MAY-23	02-JUN-23
3	3	1	1	14	26-MAY-23	02-JUN-23
4	4	1	1	17	26-MAY-23	02-JUN-23
5	5	1	1	15	26-MAY-23	02-JUN-23
6	6	2	3	2	20-APR-23	04-MAY-23
7	7	2	3	3	20-APR-23	04-MAY-23
8	8	2	3	4	20-APR-23	04-MAY-23
9	10	2	3	21	20-APR-23	04-MAY-23
10	11	2	3	22	20-APR-23	04-MAY-23
11	12	3	5	12	07-MAY-23	14-MAY-23
12	13	3	5	16	10-MAY-23	17-APR-23
13	14	3	5	5	12-MAY-23	22-APR-23
14	15	3	5	9	13-MAY-23	23-APR-23
15	16	1	9	5	24-NOV-22	08-DEC-22
16	17	1	9	6	24-NOV-22	08-DEC-22
17	18	5	9	8	04-MAR-23	11-MAR-23

-printscreen sql-developer-

## 12. Cereri SQL complexe

### Exercițiul 1:

Pentru cărțile care au fost împrumutate de mai puțin de 3 cititori să se afișeze prețul lor și numărul total de exemplare existente.

Elemente utilizate în această cerere:

- utilizarea a cel puțin 1 bloc de cerere (clauza WITH)
- subcereri sincronizate în care intervin cel puțin 3 tabele

### Rezolvare:

```
with numar_carti as(
    select id_carte, sum(nr_exemplare) nr_carti from carte_biblioteca
    group by id_carte)
select distinct(c.id_carte), c.pret, nr_carti
from carte c join imprumut imp on (c.id_carte = imp.id_carte)
    join numar_carti nc on (c.id_carte = nc.id_carte)
where 2 >= (select count(*)
    from cititor ci
    where exists(select 'x'
        from imprumut i
        where i.id_carte = c.id_carte
        and i.id_cititor = ci.id_cititor ));
```

```

138 with numar_carti as(
139     select id_carte, sum(nr_exemplare) nr_carti from carte_bibliote
140     group by id_carte)
141 select distinct(c.id_carte), c.pret, nr_carti
142 from carte c join imprumut imp on (c.id_carte = imp.id_carte)
143     join numar_carti nc on (c.id_carte = nc.id_carte)
144 where 2>= (select count(*)
145     from cititor ci
146     where exists(select 'x'
147     from imprumut i
148     where i.id_carte = c.id_carte
149     and i.id_cititor = ci.id_cititor ));

```

Explain Plan x Query Result x

SQL | All Rows Fetched: 19 in 0.131 seconds

	ID_CARTE	PRET	NR_CARTI
1	10	25	16
2	11	15	12
3	21	38	13
4	9	35	15
5	22	45	10
6	4	25	16
7	3	20	18
8	8	35	8
9	1	20	40
10	15	45	8

## Exercițiul 2:

Să se afișeze pentru fiecare bibliotecă locația pe o singura linie și numărul de împrumuturi făcute de fiecare în anii 2022 și 2023 .

Elemente utilizate în această cerere:

- subcereri nesincronizate în clauza FROM
- utilizarea a cel puțin 2 funcții pe șiruri de caractere,  
2 funcții pe date calendaristice, a cel puțin unei expresii CASE

## Rezolvare:



```

select initcap(b.denumire), concat(concat(oras, ' '),strada), an2022, an2023
from (
    select b.id_biblioteca,
        sum(case when to_char(data_imprumut,'yyyy') = 2023 then 1 else 0 end)
an2023,
        sum(case when to_char(data_imprumut,'yyyy') = 2022 then 1 else 0 end)
an2022
    from imprumut i join biblioteca b on (b.id_biblioteca = i.id_biblioteca)
    group by b.id_biblioteca) ani
join biblioteca b on (b.id_biblioteca = ani.id_biblioteca)
join locatie l on (b.id_locatie = l.id_locatie);

```

155	select initcap(b.denumire), concat(concat(oras, ' '),strada), an2022, an2023 from
156	select b.id_biblioteca,
157	sum(case when to_char(data_imprumut,'yyyy') = 2023 then 1 else 0 end) an2023
158	sum(case when to_char(data_imprumut,'yyyy') = 2022 then 1 else 0 end) an2022
159	from imprumut i join biblioteca b on (b.id_biblioteca = i.id_biblioteca)
160	group by b.id_biblioteca) ani
161	join biblioteca b on (b.id_biblioteca = ani.id_biblioteca)
162	join locatie l on (b.id_locatie = l.id_locatie);
163	
164	

INITCAP(B.DENUMIRE)	CONCAT(CONCAT(ORAS,' '),STRADA)	AN2022	AN2023
1 Biblioteca Județeană Nicolae Iorga	Ploiesti Erou Călin Cătălin 1	0	4
2 Biblioteca Națională A României	Bucuresti Bulevardul unirii 22	2	5
3 Biblioteca Județeană Dinicu Gole...	Pitesti Victoriei 18	0	1
4 Biblioteca Metropolitană București	Bucuresti Strada veteranilor ...	0	6
5 Biblioteca Județeană Octavian Goga	Cluj Mihail Kogălniceanu 12-14	0	3
6 Biblioteca Lucian Blaga	Cluj Str. Clinicilor 2	0	4
7 Biblioteca Centrală Universitară	Bucuresti Str. Boteanu nr. 1	0	4

### Exercițiul 3:

Să se afișeze detalii despre cititorii care au împrumutat cel mult 2 cărți de la editura "Corint".

Elemente utilizate în această cerere:

- grupări de date cu subcereri nesincronizate în care intervin cel puțin 3 tabele,
- funcții grup, filtrare la nivel de grupuri (în cadrul aceleiași cereri)

### Rezolvare:

```
select *
from cititor
where
id_cititor in (
    select c.id_cititor
    from cititor c
    join imprumut i on (c.id_cititor=i.id_cititor)
    join carte car on (car.id_carte = i.id_carte)
    join editura e on (e.id_editura = car.id_editura)
    where e.denumire = 'Corint'
    group by c.id_cititor
    having count(*)<3);
```

```

166 select * from cititor where
167 id_cititor in (
168     select c.id_cititor
169     from cititor c
170     join imprumut i on (c.id_cititor=i.id_cititor)
171     join carte car on (car.id_carte = i.id_carte)
172     join editura e on (e.id_editura = car.id_editura)
173     where e.denumire = 'Corint'
174     group by c.id_cititor
175     having count(*)<3);
176
177

```

Explain Plan

Query Result

SQL

All Rows Fetched: 3 in 0.006 seconds

ID_CITITOR	NUME	PRENUME	NR_TEL	DATA_NASTERII
1	1 Nae	Valentin	0720930442	20-MAY-02
2	5 Constantine...	Cristina	0732567891	06-FEB-87
3	6 Mihai	Alexandru	0745123789	12-MAR-95

## Exercițiul 4:

Să se afișeze câți cititori au împrumutat cel puțin o dată cea mai veche carte.

## Rezolvare:

```
select * from carte;
```

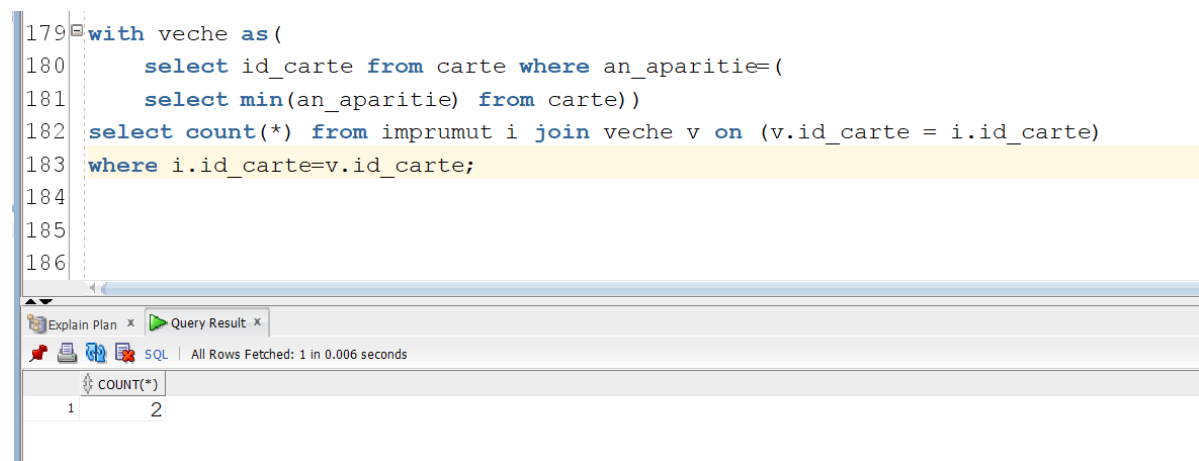
```
with veche as(
```

```
    select id_carte from carte where an_aparitie=(
    select min(an_aparitie) from carte))
```

```
select count(*) from imprumut i join veche v on (v.id_carte = i.id_carte)
```

where i.id\_carte=v.id\_carte;

```
179 with veche as (  
180     select id_carte from carte where an_aparitie=(  
181         select min(an_aparitie) from carte)  
182     select count(*) from imprumut i join veche v on (v.id_carte = i.id_carte)  
183     where i.id_carte=v.id_carte;  
184  
185  
186
```



SQL | All Rows Fetched: 1 in 0.006 seconds

COUNT(*)
2

## Exercițiul 5:

Pentru toți cititorii născuți în 2002, 2003, respectiv 2004, să se afișeze  
să se afișeze suma medie pe care aceștia au plătit-o.

Elemente utilizate în această cerere:

- ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)

## Rezolvare:

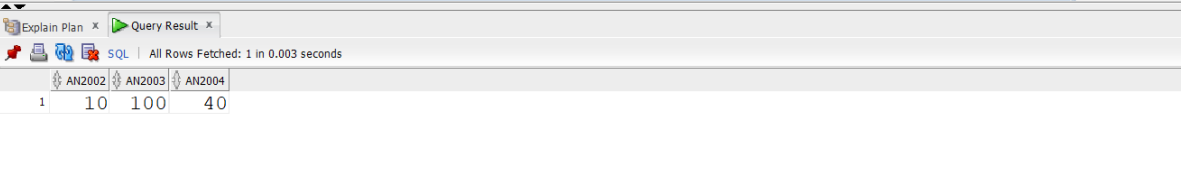
select

```
avg(decode(to_char(data_nasterii,'yyyy'), 2002, nvl(pret,0), null)) an2002,  
avg(decode(to_char(data_nasterii,'yyyy'), 2003, nvl(pret,0), null)) an2003,  
avg(decode(to_char(data_nasterii,'yyyy'), 2004, nvl(pret,0), null)) an2004  
from cititor c left join abonament a on (c.id_cititor=a.id_cititor);
```

```

186 select
187 avg(decode(to_char(data_nasterii,'yyyy'), 2002, nvl(pret,0), null)) an2002,
188 avg(decode(to_char(data_nasterii,'yyyy'), 2003, nvl(pret,0), null)) an2003,
189 avg(decode(to_char(data_nasterii,'yyyy'), 2004, nvl(pret,0), null)) an2004
190 from cititor c left join abonament a on (c.id_cititor=a.id_cititor);
191
192
193

```



	AN2002	AN2003	AN2004
1	10	100	40

## 13. Operații de actualizare și de suprimare a datelor

### Updateuri:

1) Să se mărească cu 5% salariul angajaților care lucrează ca 'Bibliotecar Sef' in bibliotecile din București.

update angajat

set salariu = salariu + salariu \* 0.05

where id\_biblioteca in (select id\_biblioteca

from biblioteca b join locatie l on(b.id\_locatie=l.id\_locatie)


where l.oras='Bucuresti')

and functie = 'Bibliotecar Sef';

```

6 update angajat
7 set salariu = salariu + salariu * 0.05
8 where id_biblioteca in (select id_biblioteca
9                          from biblioteca b join locatie l on(b.id_locatie=l.id_locatie)
10                         where l.oras='Bucuresti')
11 and functie = 'Bibliotecar Sef';
12

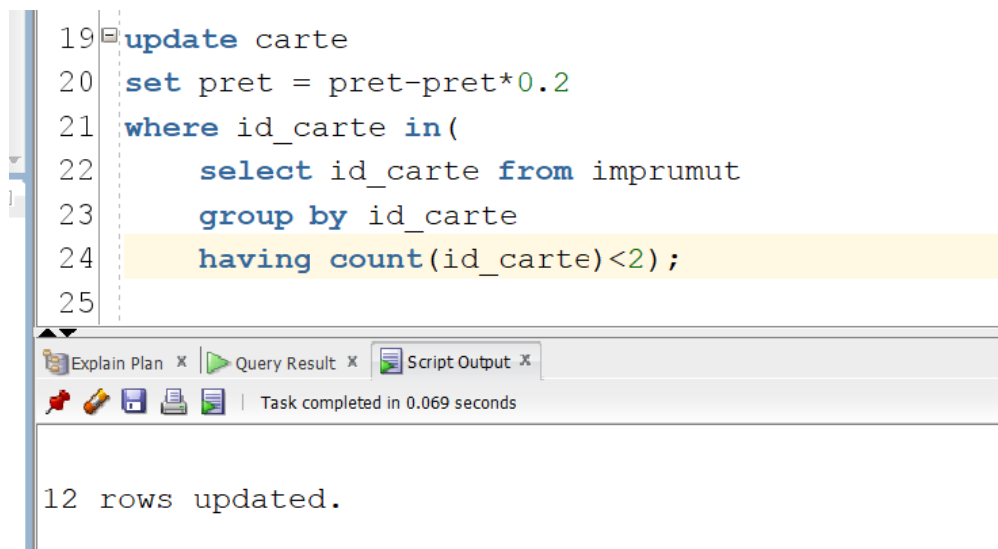
```



3 rows updated.

2) Pentru cărțile care au fost împrumutate de mai puțin de 2 ori(niciodată sau o dată) să se scadă prețul cu 20%

```
update carte
set pret = pret-pret*0.2
where id_carte in(
    select id_carte from imprumut
    group by id_carte
    having count(id_carte)<2);
```



```
19 update carte
20 set pret = pret-pret*0.2
21 where id_carte in(
22     select id_carte from imprumut
23     group by id_carte
24     having count(id_carte)<2);
25
```

Task completed in 0.069 seconds

12 rows updated.

2) Pentru cărțile care au fost împrumutate de mai puțin de 2 ori(niciodată sau o dată) să se scadă prețul cu 20%

```
update abonament
set pret = pret - pret*0.15
where id_cititor in (
    select c.id_cititor from cititor c join abonament a on (a.id_cititor =
    c.id_cititor)
    where (sysdate-data_nasterii)<6570);
```

```
29 update abonament
30 set pret = pret - pret*0.15
31 where id_cititor in (
32 select c.id_cititor from cititor c join abonament a on (a.id_cititor = c.id_cititor)
33 where (sysdate-data_nasterii)<6570);
34
35
```

Script Output x

Task completed in 0.113 seconds

1 row updated.

## Delete:

1) Să se steargă din baza de date toți cititorii care nu au un abonament.

```
delete from cititor
where id_cititor in (select c.id_cititor
                    from cititor c left join abonament a
                    on(c.id_cititor=a.id_cititor)
                    where id_abonament is null);
```

```
35
36 delete from cititor
37 where id_cititor in (select c.id_cititor
38                    from cititor c left join abonament a on(c.id_cititor=a.id_cititor)
39                    where id_abonament is null);
40
41
```

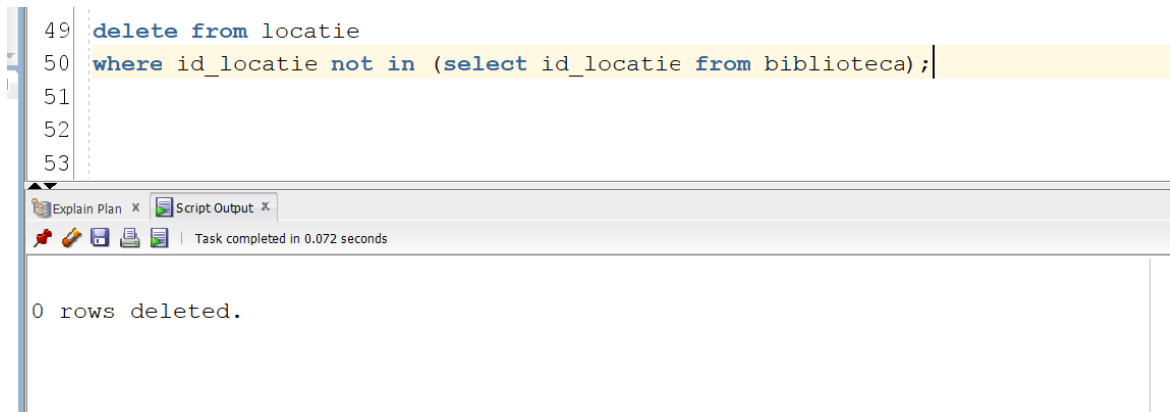
Query Result x Script Output x

Task completed in 0.076 seconds

2 rows deleted.

2) Să se steargă din baza de date locațiile cărora nu le corespund nici o bibliotecă.

```
delete from locatie
where id_locatie not in (select id_locatie from biblioteca);
```



The screenshot shows a SQL IDE with a script editor and a results pane. The script editor contains the following SQL statement:

```
49 delete from locatie
50 where id_locatie not in (select id_locatie from biblioteca);
51
52
53
```

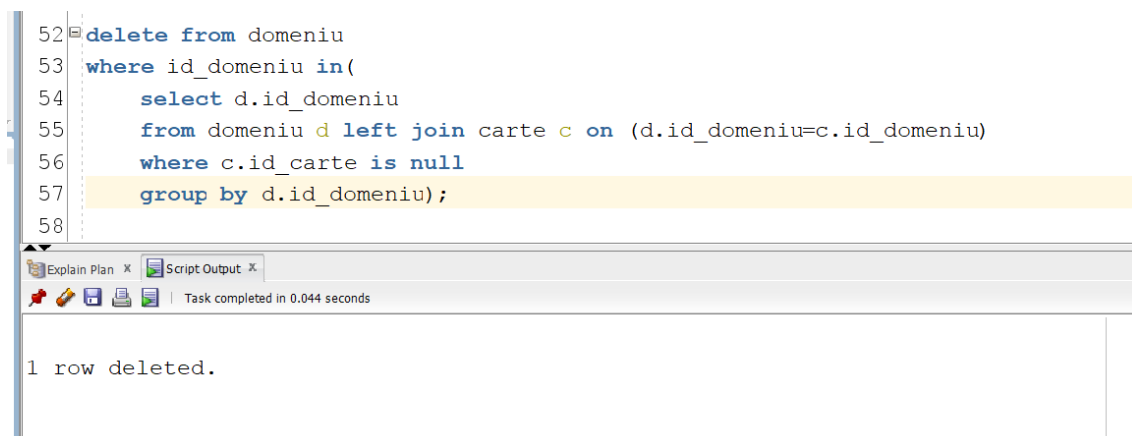
The results pane shows the output of the statement:

```
0 rows deleted.
```

The IDE interface includes tabs for 'Explain Plan' and 'Script Output', and a status bar indicating 'Task completed in 0.072 seconds'.

3) Să se steargă din baza de date domeniile cărora nu le corespund nici o carte.

```
delete from domeniu
where id_domeniu in(
    select d.id_domeniu
    from domeniu d left join carte c on (d.id_domeniu=c.id_domeniu)
    where c.id_carte is null
    group by d.id_domeniu);
```



The screenshot shows a SQL IDE with a script editor and a results pane. The script editor contains the following SQL statement:

```
52 delete from domeniu
53 where id_domeniu in(
54     select d.id_domeniu
55     from domeniu d left join carte c on (d.id_domeniu=c.id_domeniu)
56     where c.id_carte is null
57     group by d.id_domeniu);
58
```

The results pane shows the output of the statement:

```
1 row deleted.
```

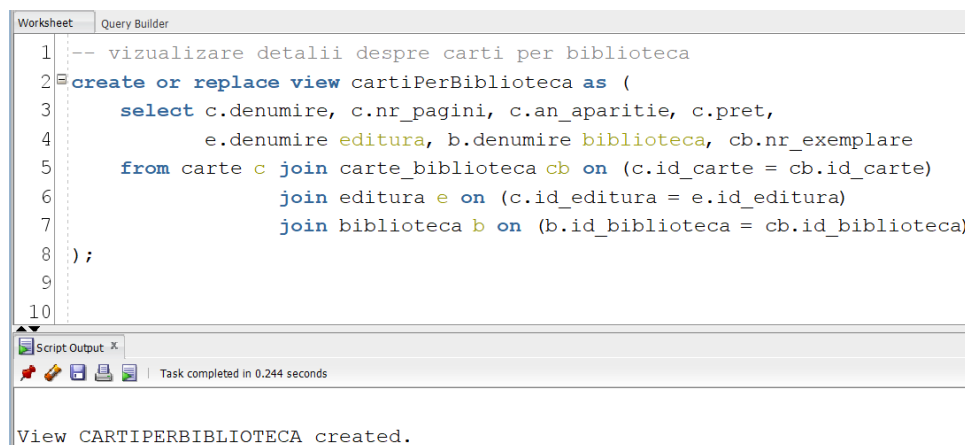
The IDE interface includes tabs for 'Explain Plan' and 'Script Output', and a status bar indicating 'Task completed in 0.044 seconds'.



## 14. Crearea unei vizualizări complexe

-- vizualizare detalii despre cărți per bibliotecă

```
create or replace view cartiPerBiblioteca as (  
    select c.denumire, c.nr_pagini, c.an_aparitie, c.pret, e.denumire  
    editura, b.denumire biblioteca, cb.nr_exemplare  
    from carte c join carte_biblioteca cb on (c.id_carte = cb.id_carte)  
        join editura e on (c.id_editura = e.id_editura)  
        join biblioteca b on (b.id_biblioteca = cb.id_biblioteca)  
);
```



-creare vizualizare-

--Exemplu de operație LMD permisă: sa se afișeze detalii despre cărțile care au exemplare la mai mult de 3 biblioteci

```
select distinct denumire, nr_pagini, an_aparitie, editura  
from cartiPerBiblioteca where denumire in (  
    select denumire from cartiPerBiblioteca  
    group by denumire  
    having count(biblioteca) > 3);
```

```

1  --Exemplu de operație LMD permisă: sa se afișeze detalii despre
2  --cărțile care au exemplare la mai mult de 3 biblioteci
3  select distinct denumire, nr_pagini, an_aparitie, editura
4  from cartiPerBiblioteca where denumire in (
5      select denumire from cartiPerBiblioteca
6      group by denumire
7      having count(biblioteca) > 3);
8
9

```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.089 seconds

	DENUMIRE	NR_PAGINI	AN_APARTIE	EDITURA
1	Tronul de cles...	512	2012	Corint
2	Hamlet	104	1603	Litera
3	Furtuna	32	1611	Litera
4	Moara cu noroc	160	1881	Litera
5	Jocul ielelor	169	1916	Paralela 45

--Exemplu de operație LMD care nu este permisă: să se modifice prețul cărții cu numele *Furtuna* cu 10%

```

update cartiPerBiblioteca
set pret = pret + pret*0.1
where denumire = 'Furtuna';

```

```

1  --Exemplu de operație LMD care nu este permisă:
2  --să se modifice prețul cărții cu numele Furtuna cu 10%
3
4  update cartiPerBiblioteca
5  set pret = pret + pret*0.1
6  where denumire = 'Furtuna';

```

Script Output x Query Result x

Task completed in 0.082 seconds

Error starting at line : 4 in command -  
 update cartiPerBiblioteca  
 set pret = pret + pret\*0.1  
 where denumire = 'Furtuna'  
 Error at Command Line : 5 Column : 5  
 Error report -  
 SQL Error: ORA-01779: cannot modify a column which maps to a non key-preserved table  
 01779. 00000 - "cannot modify a column which maps to a non key-preserved table"  
 \*Cause: An attempt was made to insert or update columns of a join view which  
 map to a non-key-preserved table.  
 \*Action: Modify the underlying base tables directly.

--Explicatie eroare: nu se poate modifica deoarece tabelul **CARTE** nu este "key preservat" in vizualizarea cartiPerBiblioteca pentru că în tabelul **CARTE** cheia primară este unică dar în vizualizare se poate repeta.

## 15. Cereri SQL folosind outer-join, division și top-n

### 1. Outer-join

-- Pentru fiecare cititor să se afișeze numărul de împrumuturi făcute  
și prețul plătit pe abonamente.

```
select nume, count(id_imprumut) as "Nr. imprumuturi", nvl(pret,0) as "Pret  
platit"
```

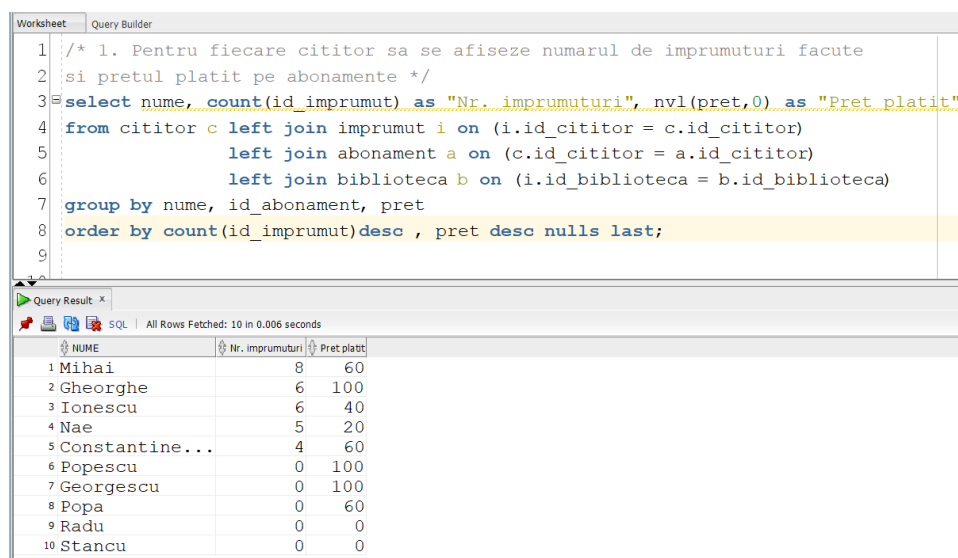
```
from cititor c left join imprumut i on (i.id_cititor = c.id_cititor)
```

```
left join abonament a on (c.id_cititor = a.id_cititor)
```

```
left join biblioteca b on (i.id_biblioteca = b.id_biblioteca)
```

```
group by nume, id_abonament, pret
```

```
order by count(id_imprumut)desc , pret desc nulls last;
```



Worksheet | Query Builder

```
1 /* 1. Pentru fiecare cititor sa se afiseze numarul de imprumuturi facute
2 si pretul platit pe abonamente */
3 select nume, count(id_imprumut) as "Nr. imprumuturi", nvl(pret,0) as "Pret platit"
4 from cititor c left join imprumut i on (i.id_cititor = c.id_cititor)
5 left join abonament a on (c.id_cititor = a.id_cititor)
6 left join biblioteca b on (i.id_biblioteca = b.id_biblioteca)
7 group by nume, id_abonament, pret
8 order by count(id_imprumut)desc , pret desc nulls last;
9
```

Query Result x

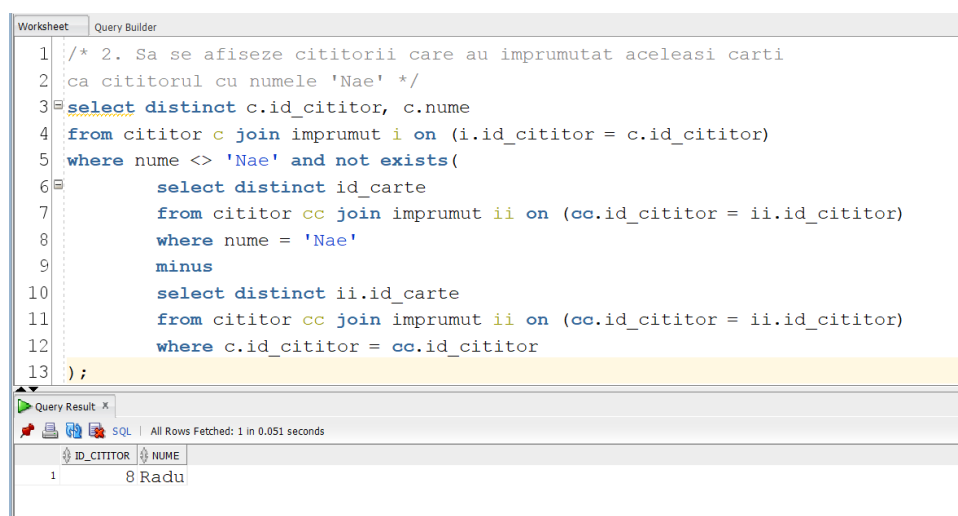
SQL | All Rows Fetched: 10 in 0.006 seconds

	NUME	Nr. imprumuturi	Pret platit
1	Mihai	8	60
2	Gheorghe	6	100
3	Ionescu	6	40
4	Nae	5	20
5	Constantine...	4	60
6	Popescu	0	100
7	Georgescu	0	100
8	Popa	0	60
9	Radu	0	0
10	Stancu	0	0

## 2. Division

--Să se afișeze cititorii care au împrumutat aceleași cărți  
ca cititorul cu numele 'Nae'

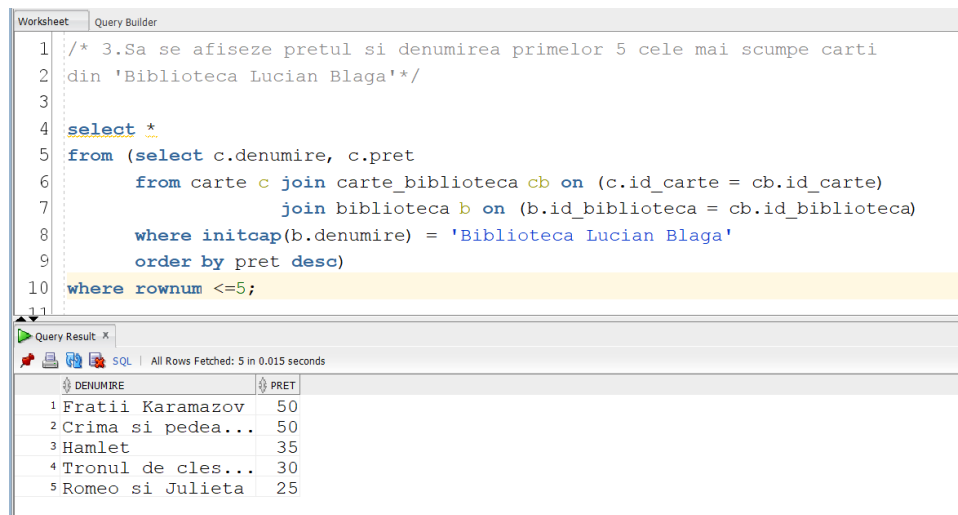
```
select distinct c.id_cititor, c.num  
from cititor c join imprumut i on (i.id_cititor = c.id_cititor)  
where nume <> 'Nae' and not exists(  
    select distinct id_carte  
    from cititor cc join imprumut ii on (cc.id_cititor = ii.id_cititor)  
    where nume = 'Nae'  
    minus  
    select distinct ii.id_carte  
    from cititor cc join imprumut ii on (cc.id_cititor = ii.id_cititor)  
    where c.id_cititor = cc.id_cititor  
);
```



## 2. Top-n

--Să se afișeze prețul si denumirea primelor 5 cele mai scumpe cărți  
din 'Biblioteca Lucian Blaga'

```
select *  
from (select c.denumire, c.pret  
      from carte c join carte_biblioteca cb on (c.id_carte = cb.id_carte)  
      join biblioteca b on (b.id_biblioteca = cb.id_biblioteca)  
      where initcap(b.denumire) = 'Biblioteca Lucian Blaga'  
      order by pret desc)  
where rownum <=5;
```



The screenshot shows a SQL query builder window with a query editor and a results pane. The query editor contains the following SQL code:

```
1 /* 3.Sa se afiseze pretul si denumirea primelor 5 cele mai scumpe carti  
2 din 'Biblioteca Lucian Blaga'*/  
3  
4 select *  
5 from (select c.denumire, c.pret  
6       from carte c join carte_biblioteca cb on (c.id_carte = cb.id_carte)  
7       join biblioteca b on (b.id_biblioteca = cb.id_biblioteca)  
8       where initcap(b.denumire) = 'Biblioteca Lucian Blaga'  
9       order by pret desc)  
10 where rownum <=5;
```

The results pane shows the following data:

	DENUMIRE	PRET
1	Fratii Karamazov	50
2	Crima si pedea...	50
3	Hamlet	35
4	Tronul de cles...	30
5	Romeo si Julieta	25

## 16. Comparația a două instrucțiuni echivalente semnatic

Fie cerința "Afișați numărul de împrumuturi realizate în anul 2023 de cititorii cu id-ul 1, 2 și 3 dacă aceștia au un abonament activ.". Vom rezolva cerința în două moduri echivalente semantic și vom explica diferențele între ele din punct de vedere al planului de execuție.

### Metoda A:

```
select c.id_cititor, c.num, count(id_imprumut)
from cititor c left join imprumut i on (c.id_cititor = i.id_cititor)
      join abonament a on (a.id_cititor = c.id_cititor)
group by c.id_cititor, c.num
having c.id_cititor in (1,2,3);
```

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				3 10
FILTER				
Filter Predicates				
OR				
C.ID_CITITOR=1				
C.ID_CITITOR=2				
C.ID_CITITOR=3				
HASH				
HASH JOIN		GROUP BY		3 10
Access Predicates		OUTER		29 9
C.ID_CITITOR=I.ID_CITITOR(+)				
MERGE JOIN				8 6
TABLE ACCESS	CITITOR	BY INDEX ROWID		10 2
INDEX	SYS_C007912	FULL SCAN		10 1
SORT		JOIN		8 4
Access Predicates				
A.ID_CITITOR=C.ID_CITITOR				
Filter Predicates				
A.ID_CITITOR=C.ID_CITITOR				
TABLE ACCESS	ABONAMENT	FULL		8 3
TABLE ACCESS	IMPRUMUT	FULL		29 3

-plan de execuție metoda A-

### Metoda B:

```

select c.id_cititor, c.num, count(id_imprumut)
from cititor c left join imprumut i on (c.id_cititor = i.id_cititor)
      join abonament a on (a.id_cititor = c.id_cititor)
group by c.id_cititor, c.num
having c.id_cititor in (1,2,3);

```

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				9
HASH		GROUP BY	1	9
HASH JOIN		OUTER	13	8
Access Predicates C.ID_CITITOR=I.ID_CITITOR(+)				
HASH JOIN		SEMI	3	5
Access Predicates C.ID_CITITOR=ID_CITITOR				
INLIST ITERATOR				
TABLE ACCESS	CITITOR	BY INDEX ROWID	3	2
INDEX	SYS_C007912	UNIQUE SCAN	3	1
Access Predicates C.ID_CITITOR=1 C.ID_CITITOR=2 C.ID_CITITOR=3				
TABLE ACCESS	ABONAMENT	FULL	3	3
Filter Predicates ID_CITITOR=1 ID_CITITOR=2 ID_CITITOR=3				
TABLE ACCESS	IMPRUMUT	FULL	13	3
Filter Predicates I.ID_CITITOR(+)=1 I.ID_CITITOR(+)=2 I.ID_CITITOR(+)=3				

-plan de execuție metoda B-

## Explicații planuri de execuție:

### Metoda A:

- folosim index după id\_cititor pentru a accesa datele din tabelul **CITITOR**;
- accesăm toate datele și din **ABONAMENT**, se sortează urmând să se facă un merge join între tabelele **ABONAMENT** și **CARTE**;

-după merge join, se accesează elementele din tabelul **ÎMPRUMUT** și se face un outer join cu rezultatul anterior după care se grupează rezultatul după informațiile dorite;

-la final, se filtrează rezultatele după cerințele dorite.

### **Metoda B:**

-folosim index după id\_cititor pentru a accesa datele din tabelul **CITITOR**;

-verificăm dacă rezultatele se află în lista (1,2,3) și le filtrăm corespunzător;

-în același timp accesăm datele din tabelul **ABONAMENT** și se face un semi hash join cu **CITITOR** după cerința dorită;

-se accesează și datele din **IMPRUMUT** și se face un outer join cu rezultatul anterior;

-la final, având toate condițiile cerute realizate deja, mai rămâne doar să grupăm și să afișăm.

Putem observa că **metoda B** este mai eficientă, rezultat care reiese și din costul mai mic prezentat în planurile de execuție. Motivul pentru care costul metodei B este mai mic, deci metoda este mai eficientă este acela că, în comparație cu **metoda A** unde accesăm toate datele necesare după care abia la final filtrăm rezultatele, aici în funcție ce accesăm un atribut, înainte de a face joinurile le și filtrăm pentru a scăpa de date nefolositoare.



## 17. a. Realizarea normalizării BCNF, FN4, FN5

### • Forma BCNF

Modelul de date proiectat se află în BCNF deoarece se află în forma normală 3 și pentru fiecare dependență funcțională  $U \rightarrow V$ ,  $U$  este o cheie candidat, în plus, fiecare determinant reprezintă o cheie candidat.

Un exemplu de tabel care n-ar fi în BCNF, ar fi următorul tabel adăugând **DATA\_PUBLICATIE**. Avem cheile candidat:

- {id\_autor, id\_carte}, {id\_autor, data\_publicatie}, {id\_carte, data\_publicatie}

Având dependența {id\_carte  $\rightarrow$  data\_publicatie} dar cum id\_carte nu este cheie candidat de una singură, tabelul nu se află în BCNF.

ID_AUTOR	ID_CARTE	DATA_PUBLICATIE
A1	C1	20
A1	C2	30
A2	C3	2
A3	C4	15

-exemplu non BCNF-

Pentru a aduce tabelul în BCNF, trebuie să împărțim tabelul în două tabele

ID_AUTOR	ID_CARTE
A1	C1
A1	C2
A2	C3

A3	C4
----	----

ID_CARTE	DATA_PUBLICATIE
C1	10/10/2022
C2	05/04/1998
C3	02/12/1985
C4	10/10/2010

-exemplu BCNF-

## • Forma FN4

Modelul de date proiectat se află în forma normală 4 deoarece se află în BCNF și nu are dependențe multi valoare. Dependența multi valoare se referă la faptul că dacă avem o dependență  $A \rightarrow B$ , dacă pentru aceeași valoare a lui A avem mai multe valori ale lui B, atunci relația reprezintă o dependență multi valoare.

Un exemplu de tabel care n-ar fi în BCNF, presupunem că am avea un tabel **ANGAJAT** cu următoarele coloane: id\_angajat, id\_biblioteca, functie; semnificând faptul că un angajat ar putea avea mai multe funcții într-o bibliotecă.

ID_ANGAJAT	ID_BIBLIOTECA	FUNCTIE
A1	B1	F1
A1	B1	F2
A2	B2	F3
A3	B3	F3

-exemplu non FN4-

Pentru a duce tabelul în forma normală 4 ar trebui să împărțim tabelul în două tabele:

ID_ANGAJAT	ID_BIBLIOTECA
A1	B1
A2	B2
A3	B3

ID_ANGAJAT	FUNCTIE
A1	F1
A1	F2
A3	F3
A4	F3

-exemplu FN4-

## • Forma FN5

O bază de date se află în forma normală 5 dacă aceasta se află deja în forma normală 4 și fiecare tabel este împărțit în cât de multe tabele posibile fără a pierde date.

Pentru a exemplifica vom lua tabelul împrumut și îl vom împărți în 3 tabele, arătând după că putem face un join între ele astfel încât să regăsim toate datele inițiale.

ID_BIBLIOTECA	ID_CITITOR	ID_CARTE
B1	C1	CA1
B1	C1	CA2
B2	C2	CA3
B3	C3	CA2

-exemplu non FN5-

ID_CARTE	ID_CITITOR	ID_BIBLIOTECA	ID_CARTE
CA1	C1	B1	CA1

CA2	C1
CA3	C2
CA2	C3

B1	CA2
B2	CA3
B3	CA2

ID_BIBLIOTECA	ID_CITITOR
B1	C1
B2	C2
B2	C3

-exemplu FN5-

Observăm că dacă facem join între primele 2 tabele după id\_cititor, obținem următorul tabel:

ID_CARTE	ID_CITITOR	ID_BIBLIOTECA
CA1	C1	B1
CA2	C1	B1
CA3	C2	B2
CA2	C3	B3

Dacă mai facem încă o dată join cu al 3-lea tabel, ajungem la tabelul de la care am plecat deci nu avem pierdere de informație, deci împărțirea a fost corectă, tabelul aflându-se acum în forma normală 5.

## 17. b. Relații/Join-uri din model reprezentate într-o bază de date nosql

Pentru această cerință vom folosi join-ul realizat între împrumut, cititor, carte, bibliotecă dorind să aflăm toate împrumuturile, din ce bibliotecă s-au realizat, ce cărți au fost împrumutate și cine le-a împrumutat cu informațiile specifice despre toate acestea. Pentru a realiza cele propuse vom folosi baza de date nosql MongoDB.

- **Creare bază de date și colecția împrumut:**

```
use proiectdb;
```

```
db.createCollection("imprumut");
```

În mongodb nu este necesar să avem o comandă specială pentru a crea o bază de date deoarece dacă aceasta nu există când încercăm să o accesăm se va crea automat.

Pentru a insera date vom folosi și comanda insertOne și comanda insertMany.

- **Inserări:**

```
db.imprumut.insertMany([
```

```
{nume:"Ionescu", prenume:"Andrei", nr_tel: "0734324442", data_nasterii: new Date("May 20, 2002"), data_imprumut: new Date("Apr 20, 2023"), data_restituire: new Date("Jun 02, 2023"), carti: ["Romeo si Julieta", "Macbeth", "Patul lui Procust", "Hamlet", "Jocul ielelor", "Povestea lui Harap-Alb"], biblioteca: "Biblioteca Nationala a Romaniei"},
```

```
{nume:"Constantin", prenume:"Cristina", nr_tel: "0756789123", data_nasterii: new Date("Feb 06, 2004"), data_imprumut: new Date("May 14, 2023"), data_restituire: new Date("Jun 02, 2023"), carti: ["Mostenitoarea Focului", "11/22/63", "1948", "Moara cu Noroc", "Tronul de clestar"], biblioteca: "Biblioteca Nationala a Romaniei"},
```

```
{nume:"Mihai", prenume:"Alexandru", nr_tel: "0745123789", data_nasterii: new Date("12 20, 1995"), data_imprumut: new Date("May 15, 2023"), data_restituire: new Date("Jun 02, 2023"), carti: ["Tronul de clestar", "Instrumente mortale", "Si soarele e o stea"], biblioteca: "Biblioteca Lucian Blaga"},
```

```
{nume:"Gheorghe", prenume:"Elena", nr_tel: "0721330579", data_nasterii: new Date("Dec 15, 1988"), data_imprumut: new Date("Nov 24, 2023"), data_restituire: new Date("Jun 02, 2023"), carti: ["Furtuna", "Hamlet"], biblioteca: "Biblioteca Centrala Universitara"},
```

```
{nume:"Popescu", prenume:"Ana", nr_tel: "0720930533", data_nasterii: new Date("May 19, 2002"), data_imprumut: new Date("Sep 16, 2023"), data_restituire: new Date("Oct 06, 2023"), carti: ["Furtuna", "Hamlet"], biblioteca: "Biblioteca Nationala a Romaniei"},
```

```
{nume:"Georgescu", prenume:"Maria", nr_tel: "0733910455", data_nasterii: new Date("Mar 12, 2000"), data_imprumut: new Date("Apr 26, 2023"), data_restituire: new Date("May 03, 2023"), carti: ["Poesii", "Ferma animalelor"], biblioteca: "Biblioteca Nationala a Romaniei"},
```

```
{nume:"Radu", prenume:"Mihnea", nr_tel: "0743922032", data_nasterii: new Date("Jun 25, 2001"), data_imprumut: new Date("Mar 13, 2023"), data_restituire: new Date("Jun 07, 2023"), carti: ["Fratii Karamazov", "Jocul ielelor", "Crima si Pedeapsa"], biblioteca: "Biblioteca Lucian Blaga"},
```

```
{nume:"Stancu", prenume:"Cristian", nr_tel: "0734314343", data_nasterii: new Date("May 14, 1998"), data_imprumut: new Date("Jun 05, 2023"), data_restituire: new Date("Jul 06, 2023"), carti: ["Stapanul umbrelor"], biblioteca: "Biblioteca Judeteană Nicolae Iorga"},
```

```
{nume:"Dumitrescu", prenume:"Andreea", nr_tel: "0755437695", data_nasterii: new Date("Sep 12, 1995"), data_imprumut: new Date("Jun 04, 2023"), data_restituire: new Date("Jun 14, 2023"), carti: ["Tronul de clestar", "Mostenitoarea focului"], biblioteca: "Biblioteca Centrala Universitara"},
{nume:"Ciobanu", prenume:"Andrei", nr_tel: "0723826392", data_nasterii: new Date("Oct 06, 1987"), data_imprumut: new Date("Nov 02, 2023"), data_restituire: new Date("Dec 22, 2023"), carti: ["Taramul de cenusa", "Statia Perzaniei"], biblioteca: "Biblioteca Nationala a Romaniei"},
{nume:"Sparda", prenume:"Vergil", nr_tel: "0749264927", data_nasterii: new Date("Nov 14, 1980"), data_imprumut: new Date("Sep 15, 2023"), data_restituire: new Date("Nov 03, 2023"), carti: ["Viseaza androizii oi electrice?", "Republica"], biblioteca: "Biblioteca Lucian Blaga"},
]);
```

- **Interogări:**

În mongodb, echivalentul selectului din sql, este functia find(). Vom da câteva exemple de interogări simple:

```
db.imprumut.find() //afiseaza tot
```

```
db.imprumut.find().sort({nume: 1}) // AFISEAZA TOT SORTAT, 1 PENTRU CRESCATOR -1 PENTRU DESCRESATOR
```

```
db.imprumut.find({nume: "Nae"}) // AFISEAZA VALOAREA CERUTA
```

```
db.imprumut.find({nume: { $eq: "Ionescu" }}) // verifica daca numele este valoarea data eq=equal
```

```
db.imprumut.find({nume: { $in: ["Ionescu", "Popescu" ]}}) //verifica daca numele se afla in lista
```

```
db.imprumut.find({ $and: [{nume:"Popescu"}, {prenume:"Ana"}]}) //Verifica mai multe conditii folosind and, daca numele e valoare si prenumele e valoare, merge si cu or
```

`db.imprumut.find({$expr: {$lt: ["$data_restituire", "$data_imprumut"]}}) // Compara 2 atribute pentru fiecare coloana, nu are sens exemplul e doar un exemplu lt=less than`

`db.imprumut.find({"carti":"Hamlet"}) // Cauta daca in vectorul de carti se afla cartea specificata`

`db.imprumut.countDocuments({data_nasterii: { $gt: new Date("Jan 01, 2000")}}) //verifica cati sunt nascuti dupa data de 1 ianuarie 2000`

- **Update:**

`db.imprumut.updateMany({nume: "Gheorghe"}, { $set: {nr_tel: "0723255173"}}) // daca numele este gheorghe, modifica numarul de telefon, merge si cu updateOne, in rest orice comanda de insert merge si pe update`

`db.imprumut.updateMany({},{$rename: {nr_tel: "numar_telefon"}}) // modifica numele unei coloane, in primele {} se poate pune si o conditie, daca in loc de rename este unset, se sterge coloana de tot`

`db.imprumut.updateOne({nume: "Gheorghe"}, {$push: { carti: "Republica"}}) // Adauga in vectorul specificat, valoarea specificata`

`db.imprumut.updateOne({nume: "Gheorghe"}, {$pull: { carti: "Republica"}}) // Sterge din vectorul specificat, valoarea specificata`

- **Delete:**

`db.imprumut.deleteOne({nume:"Ciobanu"}) // Sterge linia in functie de conditie`

## 18. Tranzacții: ilustrarea consistency levels

O tranzacție reprezintă un set de instrucțiuni asupra bazei de date folosind comenzile insert, update și delete.

Isolation: presupune execuția unei tranzacții ca și când nu există alte tranzacții care se întâmplă simultan cu aceasta.

Isolations levels:

Isolation Level	Dirty Read	Nonrepeatable Read	Phantom Read
Read uncommitted	Possible	Possible	Possible
Read committed	Not possible	Possible	Possible
Repeatable read	Not possible	Not possible	Possible
Serializable	Not possible	Not possible	Not possible

Dintre acestea, baza de date Oracle folosește doar nivelurile de izolare "Read committed" și "Serializable", implicit având nivelul "Read committed". Deci, după informațiile din tabel, "Dirty Read" este posibil doar în cazul nivelului "Read uncommitted" dar cum implicit in Oracle este "Read Committed" nu pot exista "Dirty reads" dar putem avea "Nonrepeatable Read" și "Phantom Read".

Pentru exemplificare, vom folosi modelul de date și două tranzacții simultane.

- **Non-repeatable Read**

"Non-repeatable Read" se referă la recitirea unor date care le-a citit deja înainte și observă ca altă tranzacție a modificat sau sters acele date.

Pentru a evita acest lucru, Oracle folosește "exclusive lock" adică nu lasă ca a 2-a tranzacție să modifice datele modificate de prima tranzacție până când aceasta nu se termină (commit/rollback).

Exemplu: fie tranzacțiile T1 și T2.

Inițial, înainte de nici o tranzacție avem pentru cartea "Furtuna" prețul de 20 de lei.



Worksheet

Query Builder





1

```
select * from carte where denumire='Furtuna'; -- pasul 1
```

2

Script Output

Query Result



SQL | All Rows Fetched: 1 in 0.003 seconds

ID_CARTE	DENUMIRE	NR_PAGINI	AN_APARITIE	PRET	ID_EDITURA	ID_DOMENIU
1	1 Furtuna	32	1611	20	1	2

-tranzacția T1-

Daca vrem să schimbăm prețul realizând o reducere de 10% vom folosi următorul cod:

```
update carte
set pret = pret - pret * 0.1
where denumire = 'Furtuna'; --pasul 2
```

Și observăm că rezultatul se schimbă doar în prima tranzacție, în a doua prețul având valoare inițială. Dacă acum, înainte de a finaliza prima tranzacție, încercăm să modificăm aceleași date în tranzacția T2 folosind isolation level "Read Committed", Oracle folosește automat "exclusive lock" și nu execută codul până nu încheiem prima tranzacție folosind commit/rollback.

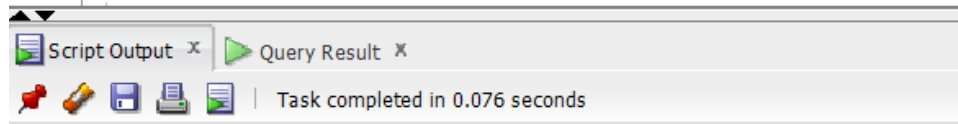
```
4  
5 update carte  
6 set pret = pret + pret * 0.1  
7 where denumire = 'Furtuna';  
8
```

The image shows a software interface with a light gray background. At the top, there are two tabs: 'Query Result' and 'Script Output', both with close buttons (X). Below the tabs is a toolbar with five icons: a green play button, a red stop button, a disk icon, a printer icon, and a document icon. To the right of the toolbar is a button labeled 'ScriptRunner Task' with a red circular icon containing a white 'X' to its right. Below this section is a large white rectangular area, likely for displaying query results or script output.

-tranzacția T2-

În momentul când facem commit primei tranzacții, se va actualiza și a doua, rezultatul ținând cont de prima tranzacție.

```
5 update carte
6 set pret = pret + pret * 0.1
7 where denumire = 'Furtuna';
8
9
```



1 row updated.

-Tranzacția T2-

Codul folosit în acest exemplu:

-Tranzacția T1:

```
select * from carte where denumire='Furtuna'; -- pasul 1
update carte
set pret = pret - pret * 0.1
where denumire = 'Furtuna'; --pasul 2
commit --pasul 5;
```

-Tranzacția T2:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED; --pasul 3
```

```
update carte
set pret = pret + pret * 0.1
where denumire = 'Furtuna'; --pasul 4
select * from carte where denumire = 'Furtuna'; --pasul 6
```

Pentru a evita "Non-repetable Read" putem folosi isolation level "Serializable" ceea ce ne va da o eroare în momentul în care vom încerca să rulăm codul din a 2-a tranzacție.

```
Error starting at line : 3 in command -
update carte
set pret = pret + pret * 0.1
where denumire = 'Furtuna'
Error report -
ORA-08177: can't serialize access for this transaction
```

Până încheiem și a doua tranzacție, indiferent dacă la prima am dat commit, datele în a doua tranzacție nu se vor modifica eliminând astfel anomalia.

Codul folosit în acest exemplu:

--Tranzacția T1:

```
select * from carte where denumire='Furtuna'; -- pasul 1
update carte
set pret = pret - pret * 0.1
where denumire = 'Furtuna'; --pasul 2
commit --pasul 5;
```

--Tranzacția T2:

```
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; --pasul 3
update carte
set pret = pret + pret * 0.1
where denumire = 'Furtuna'; --pasul 4
select * from carte where denumire = 'Furtuna'; --pasul 6
```

- **Phantom Read**

Pentru a exemplifica apariția anomaliei 'Phantom Read' vom folosi următorul cod urmând pașii precizați.

--Tranzacția T1:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED; --Pasul 1
select avg(pret) from carte; --Pasul 2
select avg(pret) from carte; --Pasul 5
commit; --Pasul 6
```

--Tranzacția T2:

```
insert into carte values(23, 'TEST', 500, 2023, 100, 4, 4); --Pasul 3
commit; --Pasul 4
```

Rulând codul urmând pașii observăm ca după ce facem commit in tranzacția T2 2, se schimbă și informațiile in tranzacția T1 chiar daca aceasta a fost începută prima, lucru pe care am vrea să îl evităm.

Pentru a evita anomalia "Phantom Read" este necesar să folosim isolation level "Serializable" ca în exemplul următor.

--Tranzacția T1:

```
insert into carte values(23, 'TEST', 500, 2023, 100, 4, 4); --Pasul 3
select avg(pret) from carte; --Pasul 4
commit; --Pasul 5
```

--Tranzacția T2:

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;	--Pasul 1
select avg(pret) from carte;	--Pasul 2
select avg(pret) from carte;	--Pasul 6
commit;	--Pasul 7

Dacă rulăm codul observăm ca până nu închidem tranzacția T2, modificările făcute de T1 nu sunt luate în considerare chiar dacă folosim commit pe tranzacția T1, eliminând astfel anomalia creată.

## 19. Optimizarea a două cereri utilizând indexare

Deoarece tabelele **CARTE** și **ÎMPRUMUT** au cele mai multe date, voi folosi câte un index pentru câte un element din fiecare pentru a face căutările mai rapide. De exemplu pentru **CARTE** voi folosi index pe *denumire*, pentru **ÎMPRUMUT** voi folosi un index pe *data\_imprumut*.

- create index idx\_carte\_denumire on carte(denumire);
- create index idx\_imprumut\_date on imprumut(data\_imprumut);

Următoarele poze reprezintă diferența în secunde dintre cererea folosind index și cererea fără index

Worksheet

Query Builder

1

2

3

```
select * from carte where denumire = 'Furtuna';
```

Explain Plan

Script Output

Query Result

SQL

All Rows Fetched: 1 in 0.01 seconds

ID_CARTE	DENUMIRE	NR_PAGINI	AN_APARITIE	PRET	ID_EDITURA	ID_DOMENIU
1	1 Furtuna	32	1611	18	1	2

-FARA INDEX-

Worksheet

Query Builder

1

select \* from carte where denumire = 'Furtuna';

2

3

create index idx\_carte\_denumire on carte(denumire);

Explain Plan

Script Output

Query Result

SQL

All Rows Fetched: 1 in 0.002 seconds

ID_CARTE	DENUMIRE	NR_PAGINI	AN_APARITIE	PRET	ID_EDITURA	ID_DOMENIU
1	1 Furtuna	32	1611	18	1	2

-CU INDEX-

**-FINAL-**